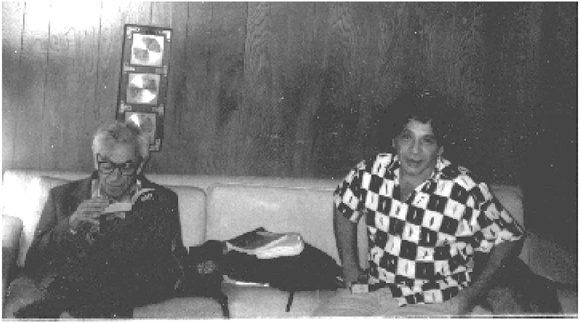
Automated Conjecturing in Mathematics - with the CONJECTURING Program

Craig Larson
(joint work with Nico Van Cleemput)

Virginia Commonwealth University
Ghent University

Waterloo ML + Security + Verification Workshop
30 August 2019

Automated Mathematical Conjecturing



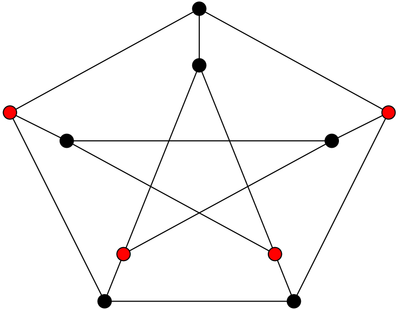
Siemion Fajtlowicz (r.)—developer of GRAFITTI

Two Kinds of Mathematical Problem

1. Finding an invariant (feature) value of a specific mathematical object.

2. Determining the class membership of a mathematical object (classification).

1^{st} Example: Independence Number of a Graph

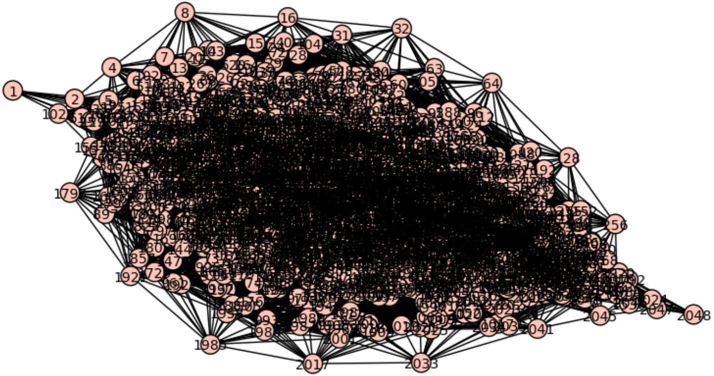


• The independence number α of a graph is the largest number of mutually non-adjacent vertices.

$$\alpha = 4.$$

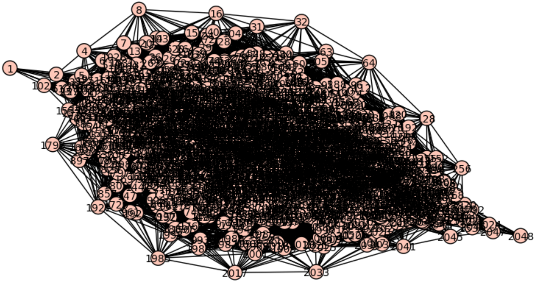
An Open Problem

What is the independence number of Sloane's DC2048?



ML Idea

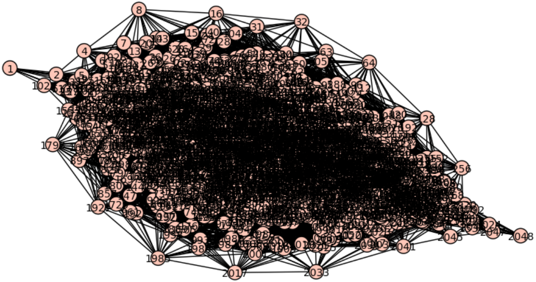
Use the values of the independence number and other invariants (features) of other (smaller) graphs as inputs to a neural net, etc.



Efficiently computable graph invariants: order, size, matching number, radius, diameter, etc.

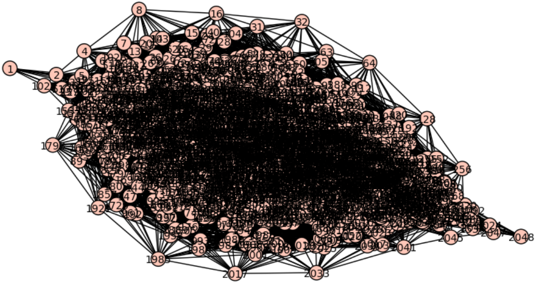
ML Idea

Problem: for many mathematical purposes you want a proof (that the invariant calculation is correct)



A Complementary Approach

Instead of the number itself, find theoretical bounds for the number that can be used to both predict and verify the number.



Inputs to our CONJECTURING program

Graph examples and corresponding values for the independence number (or other invariant of interest) and other (efficiently computable) invariant values.

Essentially, the input is a data table consisting of graphs and invariant values.

Outputs of our CONJECTURING program

- The output is expressions representing conjectured (upper or lower) bounds for the independence number of a graph.

Outputs of our CONJECTURING program

- The output is expressions representing conjectured (upper or lower) bounds for the independence number of a graph.
- These may be true—but would then need to be proved.

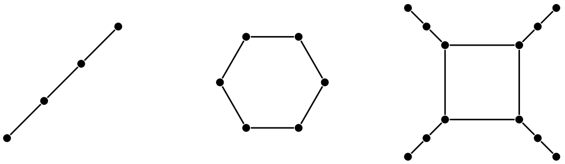
Outputs of our CONJECTURING program

- The output is expressions representing conjectured (upper or lower) bounds for the independence number of a graph.
- These may be true—but would then need to be proved.
- These may be false—in which case there is a counterexample.

A Conjectured Upper Bound Theorem

Theorem

For any connected graph, $\alpha \leq$ order $-$ radius.



r-ciliates: $C_{1,1}$, $C_{3,0}$, $C_{2,2}$

The Heuristic

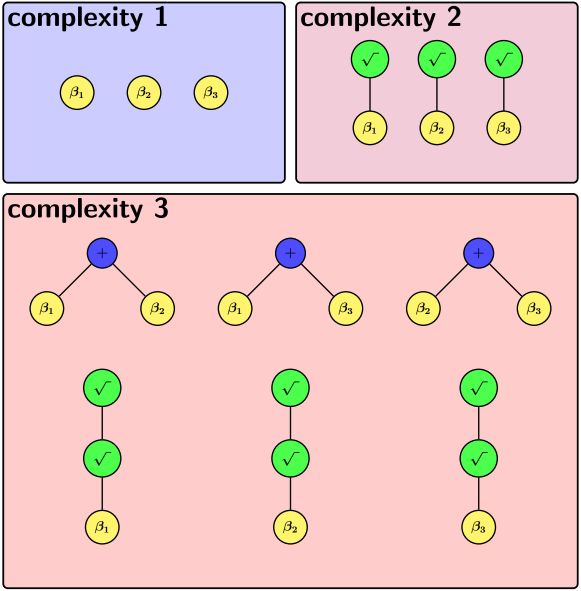
Generate all syntactically possible expressions starting from smallest complexity, and apply it to the data table: check for truth and significance.

The Heuristic

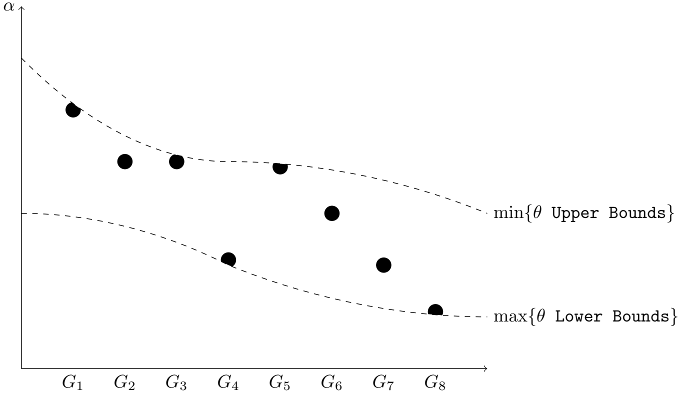
Generate all syntactically possible expressions starting from smallest complexity, and apply it to the data table: check for truth and significance.

1. (Truth) The bound must be true for all of the objects in the data table.
2. (Significance) The bound must be better than every stored bound for at least one object in the data table.

Generating Possible Bounds for an Invariant



The Heuristic



The Heuristic is General

Any (mathematical) object and invariant (feature) of that object.

- Graphs
- Matrices
- Integers
- Combinatorial Games
- Intersecting Set Systems

Matrix Theory—Determinants of Symmetric Matrices

determinant(x) $\leq$ permanent(x)
determinant(x) $\leq$ maximum_eigenvalue(x)*trace(x)
determinant(x) $\leq$ (rank(x) + 1)*spectral_radius(x)

determinant(x) $\geq$ minimum_eigenvalue(x)*separator(x)
determinant(x) $\geq$ minimum(permanent(x), log(nullity(x)))

Number Theory—Goldbach's Conjecture

Conjecture: Any even number greater than 2 can be written as a sum of two primes.

Number Theory—Goldbach's Conjecture

Conjecture: Any even number greater than 2 can be written as a sum of two primes.

For any even integer $x > 2$ let Goldbach(x) be the number of ways x can be written as a sum of two primes.

Number Theory—Goldbach's Conjecture

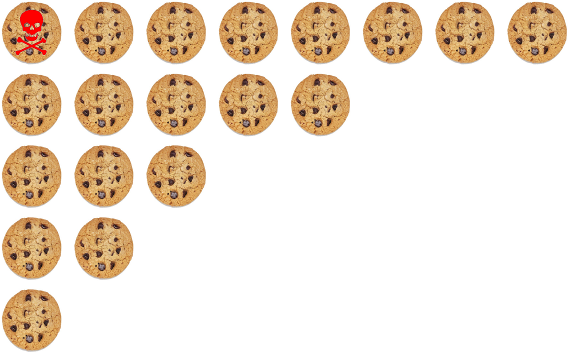
Conjecture: Any even number greater than 2 can be written as a sum of two primes.

For any even integer $x > 2$ let Goldbach(x) be the number of ways x can be written as a sum of two primes.

Conjecture: Goldbach(x) $\geq$ digits10(x) - 1

Def. *digits10(x)* is the number of digits in the base-10 representation of x.

Bounds for Chomp invariants



Bounds for Chomp invariants

Conjectured Theorem:

For any position where the *previous-player-to-play* has a winning strategy (a *P*-position),

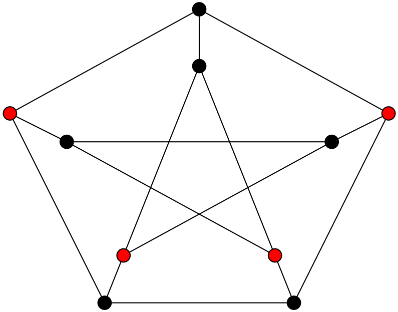
number of cookies on the board $\geq$ 2* the number of columns -1.

Theory

It is possible to tell the program theoretical knowledge—and require it to improve on that knowledge.

Theory

It is possible to tell the program theoretical knowledge—and require it to improve on that knowledge.



The THEORY parameter

- We want conjectures that are not implied by existing theory (theoretical bounds, known bounds, “auxilliary truths”),
- that is, conjectures that give a better bound for at least one graph,
- so, for us, at least one graph in our data table.
- This the `theory` parameter in the program's function call.

Some Lower Bounds for Independence

- $\alpha \geq$ radius.
- $\alpha \geq$ residue.
- $\alpha \geq$ critical independence number
- $\alpha \geq$ `max_even_minus_even_horizontal`

An Open Lower Bound Conjecture

$\alpha \geq$ `min(girth, floor(lovasz_theta))`

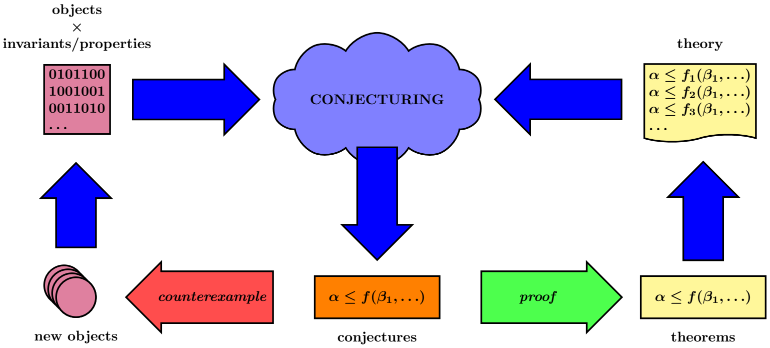
Equivalently, $\alpha \geq$ `girth` or $\alpha =$ `floor(lovasz_theta)`

Some Upper Bounds for Independence

- $\alpha \leq$ annihilation number
- $\alpha \leq$ fractional independence number
- $\alpha \leq$ Lovász number
- $\alpha \leq$ Cvetković bound
- $\alpha \leq$ order - matching number.
- $\alpha \leq$ Hansen-Zheng bound.

(The *Hansen-Zheng bound* is
$\lfloor \frac{1}{2} + \sqrt{\frac{1}{4} + \text{order}^2 - \text{order} - 2 \cdot \text{size}} \rfloor$.)

The CONJECTURING Process



Kinds of Knowledge

1. Concepts: invariants, properties: these will be columns in the data table.

Kinds of Knowledge

1. Concepts: invariants, properties: these will be columns in the data table.

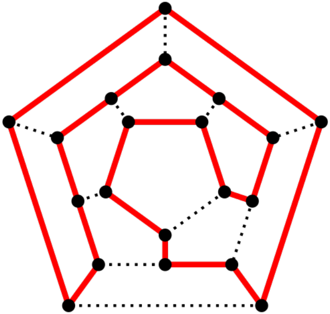
2. Theorems. These can be used for verifying ML outputs.

Kinds of Knowledge

1. Concepts: invariants, properties: these will be columns in the data table.
2. Theorems. These can be used for verifying ML outputs.
3. Counterexamples. These can be fed back into the CONJECTURING program.

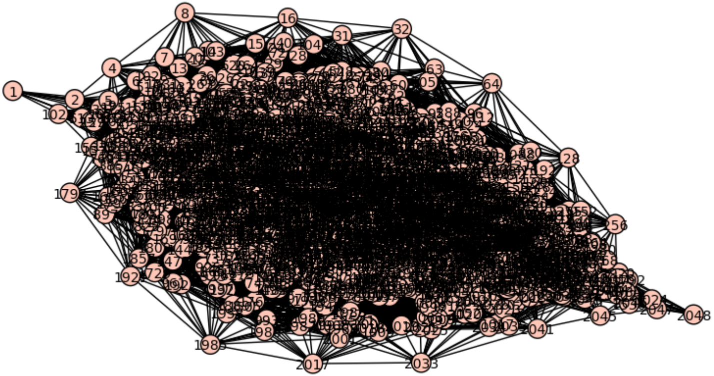
2^{nd} Example: Graph Hamiltonicity

A Hamiltonian cycle in a graph is a cycle that covers all of the vertices of the graph.



A Problem

Is Sloane's DC2048 hamiltonian?



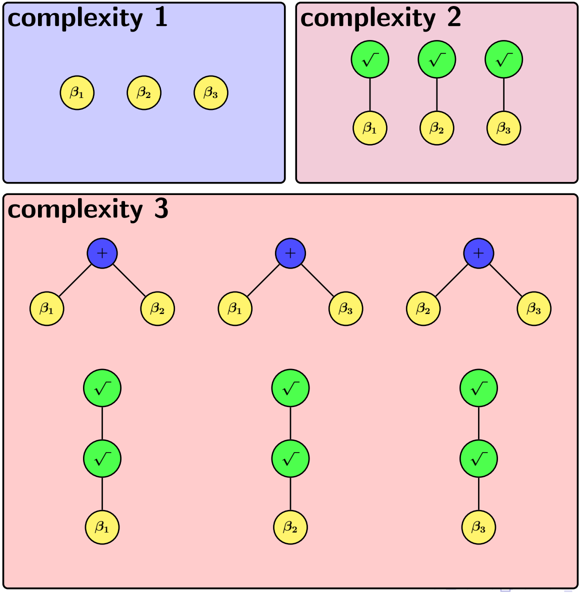
2^{nd} Example: Graph Hamiltonicity

Basic Problem: does a specific graph belong to a certain class?

Input: a data table of graphs and computed properties.

Output: Expressions representing sufficient conditions for a graph to have a property.

Generating Possible Sufficient Conditions for Class Membership



Necessary Conditions for Hamiltonicity

- If a graph is hamiltonian then it is 2-connected.

- If a graph is hamiltonian then it is *van den heuvel* (Laplacian eigenvalues condition).

Sufficient Conditions for Hamiltonicity

(Dirac) If the minimum degree of a graph is at least half the order then the graph is hamiltonian.

(Ore) If the sum of the degrees of any pair of non-adjacent vertices is at least n then the graph is hamiltonian.

(Chvatal-Erdős) If the vertex connectivity of a graph is at least the independence number then the graph is hamiltonian.

Three Conjectured Theorems

Def. A graph is class 1 is its edges can be properly colored with the number of colors of the maximum degree of any of its vertices.

Thm. `(is_hamiltonian)->((is_cubic)->(is_class1))`

If is a graph is hamiltonian then if it is cubic it is hamiltonian.

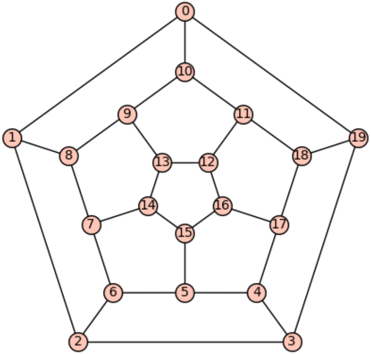
If a graph is hamiltonian then either it is not cubic or it is class 1.

If a graph is hamiltonian and cubic then it is class 1.

Three Conjectured Theorems

Thm. (is_planar_transitive)->(is_hamiltonian)

If a graph is planar and vertex-transitive then it is hamiltonian.



Three Conjectured Theorems

Thm. `(is_planar_transitive)->(is_hamiltonian)`

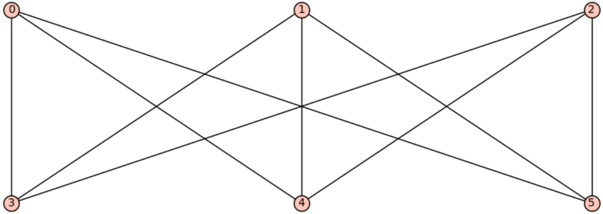
If a graph is planar and vertex-transitive then it is hamiltonian.

1. Every vertex-transitive graph is regular.
2. (Mader, 1970) If a graph is d-regular vertex-transitive with connectivity κ then $\frac{2(d+1)}{3} \leq \kappa$.
3. (Tutte, 1956) Every 4-connected planar graph is Hamiltonian.
4. (Zelinka, 1977) If a graph is planar, vertex-transitive and 3-regular then it is one of 8 specific graphs or an n-sided prism.
5. Only need to check the prisms!

Three Conjectured Theorems

Thm. ((is_bipartite) & (is_strongly_regular))->(is_hamiltonian)

If a graph is bipartite and strongly regular then it is hamiltonian.



The Heuristic is General

Any (mathematical) object and invariant (feature) of that object.

- Graph Hamiltonicity
- Graph Pebbling
- Bootstrap Percolation
- Matrices, Integers, Combinatorial Games, etc.

The Heuristic is General—SAT Idea

Any (mathematical) object and invariant (feature) of that object.

- Objects: boolean formulas.

The Heuristic is General—SAT Idea

Any (mathematical) object and invariant (feature) of that object.

- Objects: boolean formulas.
- Properties: Satisfiable, unsatisfiable

The Heuristic is General—SAT Idea

Any (mathematical) object and invariant (feature) of that object.

- Objects: boolean formulas.
- Properties: Satisfiable, unsatisfiable
- Define lots of efficiently-checkable boolean formula properties.

The Heuristic is General—SAT Idea

Any (mathematical) object and invariant (feature) of that object.

- Objects: boolean formulas.
- Properties: Satisfiable, unsatisfiable
- Define lots of efficiently-checkable boolean formula properties.
- Input: data table of boolean formulas and property values.

The Heuristic is General—SAT Idea

Any (mathematical) object and invariant (feature) of that object.

- Objects: boolean formulas.
- Properties: Satisfiable, unsatisfiable
- Define lots of efficiently-checkable boolean formula properties.
- Input: data table of boolean formulas and property values.
- Output: conjectured sufficient conditions for satisfiability of boolean formulas.

The Heuristic is General—SAT Idea

Any (mathematical) object and invariant (feature) of that object.

- Objects: boolean formulas.
- Properties: Satisfiable, unsatisfiable
- Define lots of efficiently-checkable boolean formula properties.
- Input: data table of boolean formulas and property values.
- Output: conjectured sufficient conditions for satisfiability of boolean formulas.
- These conjectures are true or false.

The Heuristic is General—SAT Idea

Any (mathematical) object and invariant (feature) of that object.

- Objects: boolean formulas.
- Properties: Satisfiable, unsatisfiable
- Define lots of efficiently-checkable boolean formula properties.
- Input: data table of boolean formulas and property values.
- Output: conjectured sufficient conditions for satisfiability of boolean formulas.
- These conjectures are true or false.
- If true, they can be proved and *used*.

The Heuristic is General—SAT Idea

Any (mathematical) object and invariant (feature) of that object.

- Objects: boolean formulas.
- Properties: Satisfiable, unsatisfiable
- Define lots of efficiently-checkable boolean formula properties.
- Input: data table of boolean formulas and property values.
- Output: conjectured sufficient conditions for satisfiability of boolean formulas.
- These conjectures are true or false.
- If true, they can be proved and *used*.
- If false, there are counterexamples.

The Heuristic is General—SAT Idea

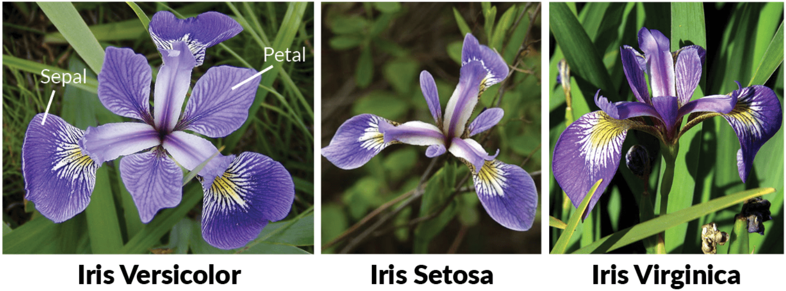
Any (mathematical) object and invariant (feature) of that object.

- Objects: boolean formulas.
- Properties: Satisfiable, unsatisfiable
- Define lots of efficiently-checkable boolean formula properties.
- Input: data table of boolean formulas and property values.
- Output: conjectured sufficient conditions for satisfiability of boolean formulas.
- These conjectures are true or false.
- If true, they can be proved and *used*.
- If false, there are counterexamples.
- Maybe there is a possibility towards a mathematical theory of satisfiable formulas that can be developed in support of and in conjunction with ML researchers?

Special Properties of Mathematical Objects

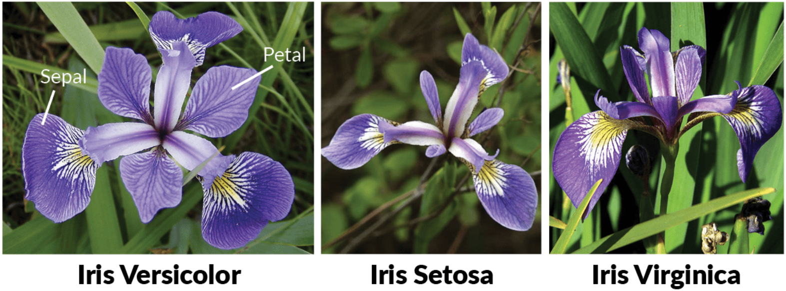
- No noise
- Knowledge is built up from concept definitions (sharp boundaries),
- Potential counterexamples can be generated from concept definitions.

A Non-mathematical Experiment



Problem: Classifying Iris examples into 3 species: 50 examples of each of *iris viginica*, *iris setosa*, and *iris versicolor*, and 4 features of each: petal length and width, and sepal length and width.

A Non-mathematical Experiment



Main Idea: generating properties from invariants by conjecturing necessary conditions for all test examples, and then using these for generating sufficient conditions for class membership.

Iris Setosa Classification Conjectures

```
(sepal_length(x) <=sepal_width(x)+2)->(is_iris_setosa(x))
```

- True for 24 of the 25 test setosa examples (96%).
- True for 0 of 50 non-setosa test examples (0% false positives).

Iris Versicolor Classification Conjectures

```
(~(sepal_length<=petal_width^(2*petal_length-2*sepal_width-1)))

->(is_iris_versicolor(x))
```

- True for 25 of 25 versicolor test examples (100%)
- True for 5 of 50 non-versicolor test examples (10% false positives).

Iris Virginica Classification Conjectures

```
(~(sepal_length<=(petal_width+1)/(log(petal_length)/log(10))^sep
 ->(is_iris_virginica(x))

(~(sepal_length>=log(floor(sepal_width))+petal_length))
 ->(is_iris_virginica(x))

(~(sepal_length<=floor(petal_length/petal_width)+sepal_width+1))
 ->(is_iris_virginica(x))

(~(sepal_length<=2*ceil(sepal_width)-petal_width+2))
 ->(is_iris_virginica(x))

(~(sepal_length<=(petal_length/petal_width)^sqrt(ceil(sepal_widt
 ->(is_iris_virginica(x))
```

- True for 24 of 25 test virginica examples (96% success).
- True for 3 of 50 non-virginica test examples (6% were false positives).

Auxiliary Truths Outside of Mathematics?

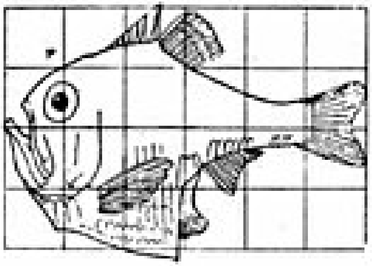


Fig. 517. *Argyropelecus Olfersi.*

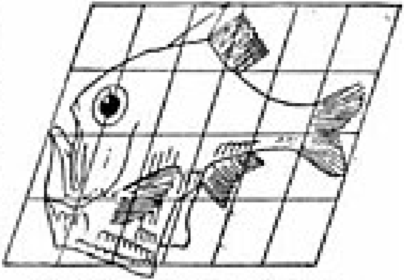


Fig. 518. *Sternoptyx diaphana.*

From: D'Arcy Thompson, *On Growth and Form*, 1917/1942.

Auxiliary Truths Outside of Mathematics?

- Auxiliary truths can be useful—to constrain search.
- We know not everything is in the data.
- Improvement may involve going back-and-forth with domain experts—they know lots of relevant concepts.

Graph Brain Project

- 112 efficiently computable properties, 36 intractable properties.
- 585+ graphs (and various collections: Sloane, DIMACS, pebbling)
- 127 efficiently computable invariants, and 33 intractable invariants.
- Database of values of (most of) these.

Thank You!

Automated Conjecturing in Sage:
`nvcleemp.github.io/conjecturing/`

Graph Brain Project:
`github.com/math1um/objects-invariants-properties`

clarson@vcu.edu