How $\underset{\text{verb, gerund}}{\underline{\hspace{3cm}}}$ the $\underset{\text{noun}}{\underline{\hspace{3cm}}}$ affects the $\underset{\text{adjective}}{\underline{\hspace{3cm}}}$ $\underset{\text{property}}{\underline{\hspace{3cm}}}$ of $\underset{\text{proper noun}}{\underline{\hspace{3cm}}}$ $\underset{\text{collection, plural}}{\underline{\hspace{3cm}}}$.

[redacted] [redacted]
[redacted]
{[redacted], [redacted]}@[redacted]

Abstract

Word Blanks is a game in which one player has as input a secret paragraph with some words replaced with blanks with part-of-speech tags, such as $\underset{\text{plural noun}}{\underline{\hspace{2cm}}}$, and the other player has as input the experience of being human for long enough to understand parts-of-speech. The first player queries the other player with part-of-speech tags and receives words to put in the corresponding blanks. The goal of Word Blanks is to maximize the humor of the resulting paragraph.

First, we collected data from a representative population of 11 theoretical computer science grad students as the second player in the game, then chose 5 filled paragraphs and asked 20 computer science grad students which of a pair was funnier given the order they were presented.

Second, we took our data and used a logistic regression model to learn which paragraph in any ordered pair is more humorous, and applied the model to the other filled paragraphs.

Our results are that humor dominance is a strict partial order, and that the second paragraph presented is usually the most humorous.

1. Introduction

Formally, Word Blanks is a family of functions $\mathcal{WB} = \{\mathsf{wb}_k\}_{k\in\mathbb{N}}$ each taking a secret paragraph p and an response function r and returning a filled paragraph in the human experience monad $\mathbb{H}$.

$$\begin{aligned} p &:: \P = \mathsf{PoS}^k \times (\mathsf{W}^k \to \mathbb{H}\,\mathsf{W}^n), k \leq n \\ r &:: \mathfrak{R} = \mathsf{PoS}^k \to \mathbb{H}\,\mathsf{W}^k \\ \mathsf{wb}_k &:: \P \to \mathfrak{R} \to \mathbb{H}\,\mathsf{W}^n \\ \mathsf{wb}_k\; p\; r &= r\; p_0 >>= p_1 \end{aligned}$$

DOI: http://dx.doi.org/10.1145/(to come)

Tag	Part of Speech	Your Word
JJ	ordinal adjective or numeral	
NN	noun	
NNS	plural noun	
JJ	ordinal adjective or numeral	
NNP	singular proper noun	
NNS	plural noun	
VBN	past participle verb	
NNP	singular proper noun	
VBD	past tense verb	
NNS	plural noun	
NN	noun	
VBN	past participle verb	
JJ	ordinal adjective or numeral	
VBZ	3rd p. sing. present tense verb	

Table 1. Input format

We investigated the Word Blanks humor relation $\hbar$, which takes two filled-in paragraphs and determines which one is funnier in $\mathbb{H}$.

$$\hbar :: \mathsf{W}^n \to \mathsf{W}^n \to \mathbb{H}\;2$$

2. The Text

Given our audience, we secretly chose our paragraph as the abstract of a notable theoretical computer science paper.

2.1 Original Quote

”A large class of computational problems involve the determination of properties of graphs, digraphs, integers, arrays of integers, finite families of finite sets, boolean formulas and elements of other countable domains. Through simple encodings from such domains into the set of words over a finite alphabet these problems can be converted into language recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily solved when an algorithm for its solution is found which terminates within a number of steps bounded by a polynomial in the length of the input. We show that a large number of classic unsolved problems of covering, matching, packing, routing, assignment and sequencing are equivalent, in the sense that either each of them possesses a polynomial-bounded algorithm or none of them does.” [4]

2.2 The Word Blanks Game

We used `nltk`, Python's Natural Language Tool Kit [3], to generate part-of-speech tags. Due to its mistakes, the paragraph was modified slightly as follows:

The $\underset{\text{ordinal adjective or numeral}}{\underline{\hspace{6em}}}$ class of computational problems involve the $\underset{\text{noun}}{\underline{\hspace{4em}}}$ properties of graphs, digraphs, integers, $\underset{\text{plural noun}}{\underline{\hspace{3em}}}$ of integers, finite families of $\underset{\text{ordinal adjective or numeral}}{\underline{\hspace{6em}}}$ sets, $\underset{\text{singular proper noun}}{\underline{\hspace{5em}}}$ -ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of $\underset{\text{plural noun}}{\underline{\hspace{3em}}}$ over a finite alphabet these problems can be $\underset{\text{past participle verb}}{\underline{\hspace{4em}}}$ into $\underset{\text{singular proper noun}}{\underline{\hspace{5em}}}$ recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily $\underset{\text{past tense verb}}{\underline{\hspace{4em}}}$ when an algorithm for its solution is found which terminates within a number of $\underset{\text{plural noun}}{\underline{\hspace{3em}}}$ bounded by a polynomial in the $\underset{\text{noun}}{\underline{\hspace{2em}}}$ of the input. We show that a large number of classic $\underset{\text{past participle verb}}{\underline{\hspace{4em}}}$ problems of covering, matching, packing, routing, assignment and sequencing are $\underset{\text{ordinal adjective or numeral}}{\underline{\hspace{6em}}}$ -order equivalent, in the sense that either each of them $\underset{\text{3rd p. sing. present tense verb}}{\underline{\hspace{7em}}}$ a polynomial - bounded algorithm or none of them do .

3. Procedure

In the first part of the experiment, data sources were recruited at Theory Tea and asked to fill in copies of Table 2.1 printed on slips of paper. The sources were not told what their input would be used for, and we replaced their names with NP-Complete problems for anonymity. 11 sets of words were collected in total.

Three other sets of words were also used, which were generated as follows:

- One set was randomly generated using the lists made available at [1].
- One set was generated using the next word prediction features of the Messaging app Version 4.4.2-G730AUCUBNG4 on an Android cellular phone.
- One set was the actual set of words used in [4].

The sets of words thusly obtained were used to complete the Word Blanks game from Section 2.2 to generate test paragraphs. Out of these, a set of 5 test paragraphs were selected according to the whims of the authors.

In the next part of the experiment, a random set of 20 evaluators were chosen based on their proximity to [redacted]. After an apology for the rude interruption from their research, each was presented with two test paragraphs, one at a time, and asked to read them without being told what for. After the evaluator had read both paragraphs, they were asked which of the two they found funnier. For each evaluator, the identity of the paragrphs presented, the order in which they were presented, and the identity of the one judged funnier were recorded. From 20 evaluators, we had one comparison for each pair of paragraphs in either order of presentation.

Once this data was obtained, Logistic Regression (specifically, the implementation in the scikit-learn Python package [2]) was used to predict the results of comparisons between the remaining pairs. The input data for the regression was generated as follows:

1. Each character in each word was converted to its ASCII value.
2. For each blank, all words corresponding to that blank across all sets of words were padded with zeros to be of the same length.
3. For each set, these padded ASCII words were concatenated to yield one vector of numbers.
4. For each pair of sets, their vectors were concatenated to yield the final input vector.

The set of evaluations obtained from the evaluators was then used as training data to the logistic regression-based classifier, which was then used to predict results of comparisons between all pairs of sets of words.

4. Results

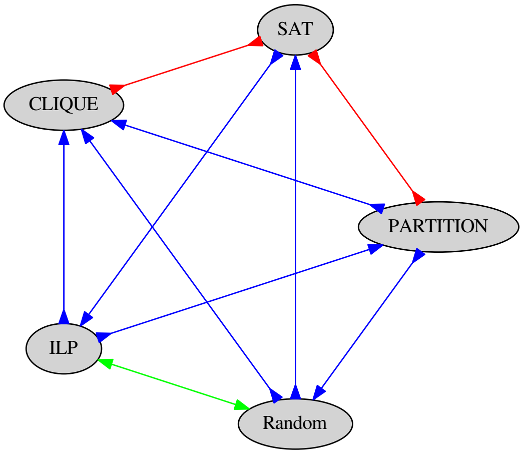

Figure 1. Graphical representation of collected evaluations

The results from the evaluators is presented in Tables 2 and 3. The test paragraphs used are included in Appendix A.1. The same is represented graphically in Figure 1. In all diagrams and tables, blue represents that one paragraph in the pair was funnier than the other whether it was presented first or second; green represents that the first paragraph presented was always funnier, and red that the second paragraph presented was always funnier. In the graphs, blue arrows have both ends pointing in the same direction (away from the funnier paragraph), green arrows point outward, and red inward.

The following interesting observations may be drawn from our experimental data:

- 75% of theorists are more funny when they go second, whereas random won most when it went first.
- **ILP** (a theorist) is isomorphic to **Random**.
- The two best performing theorists both referenced U.S. presidents or presidential candidates.

	Random	PARTITION	CLIQUE	SAT	ILP
Random		▲	◀	◀	◁
PARTITION	◀		◀	△	▲
CLIQUE	▲	▲		△	▲
SAT	▲	△	△		◀
ILP	◁	◀	◀	▲	

Table 2. Collected evaluations. The paragraph corresponding to the author on the left was shown first. Arrows point to the winner in each comparison.

		Random	PARTITION	CLIQUE	SAT	ILP
Wins as	**1st**	3	2	0	1	3
Wins as	**2nd**	2	3	1	3	2
Wins	**total**	5	5	1	4	5

Table 3. Number of victories when presented first, second, and in total, from collected evaluations

- The two best performing theorists both used concrete numbers as their ”-order equivalent” word
- 42 was put by **SAT** and **ILP** in the same blank.

The results of the predictions of the model is represented in Table 4 tabularly and Figure 3 graphically, whereas Figure 2 contains just the blue ”dominance” arrows. Our striking result is that humor dominance is a strict partial order, meaning it is anti-reflexive and transitive.

When considering how *varying* the *ordering* affects the *relative humor* of *Word Game pairs*, we noticed that our model predicted 21 blue arrows, 22 green arrows, and 51 red arrows, thus that the number of second-place victories is $\frac{123}{188} = 0.6542553191489362\%$, compared to a mere $\frac{65}{188} = 0.3457446808510638 6\%$ first-place victories. Thus, a paragraph is nearly 1.8923076923076922 times more likely to be more humorous if it appears second.

5. Acknowledgements

We would like to thank our test subject theorists for volunteering their human experience to science: G, Will, Nick, Adam, Aviv, Sam, Shalev, Pritish, Rio, Prashant, and Atalay.

References

[1] Randomlists.com. http://www.randomlists.com/. Accessed: 2016-02-29.

[2] scikit-learn. http://scikit-learn.org/. Accessed: 2016-02-29.

[3] S. Bird, E. Klein, and E. Loper. *Natural Language Processing with Python*. O’Reilly Media, 2009.

[4] R. M. Karp. Reducibility among combinatorial problems. In *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York.*, pages 85–103, 1972. URL http://www.cs.berkeley.edu/~luca/cs172/karp.pdf.

A. Paragraphs

A.1 Test Paragraphs

These were the paragraphs actually presented to evaluators.

1. **Random:**

   The eigth class of computational problems involve the paper properties of graphs, digraphs, integers, tables of integers, finite families of two sets, Freddie Mercury-ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of noses over a finite alphabet these problems can be tickled into Robert Downey Jr. recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily robbed when an algorithm for its solution is found which terminates within a number of arches bounded by a polynomial in the crowd of the input. We show that a large number of classic subtracted problems of covering, matching, packing, routing, assignment and sequencing are last-order equivalent, in the sense that either each of them jumps a polynomial-bounded algorithm or none of them do.

2. **PARTITION:**

   The double class of computational problems involve the bathroom properties of graphs, digraphs, integers, stars of integers, finite families of sixteen sets, Barack-ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of shoes over a finite alphabet these problems can be eaten into Ara Gorn recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily barfed when an algorithm for its solution is found which terminates within a number of swamps bounded by a polynomial in the buttox of the input. We show that a large number of classic holpen problems of covering, matching, packing, routing, assignment and sequencing are 32,426,941-order equivalent, in the sense that either each of them beats a polynomial-bounded algorithm or none of them do.

3. **CLIQUE:**

   The nth class of computational problems involve the wolverine properties of graphs, digraphs, integers, wolverines of integers, finite families of i sets, [redacted]-ean formulas and elements of other

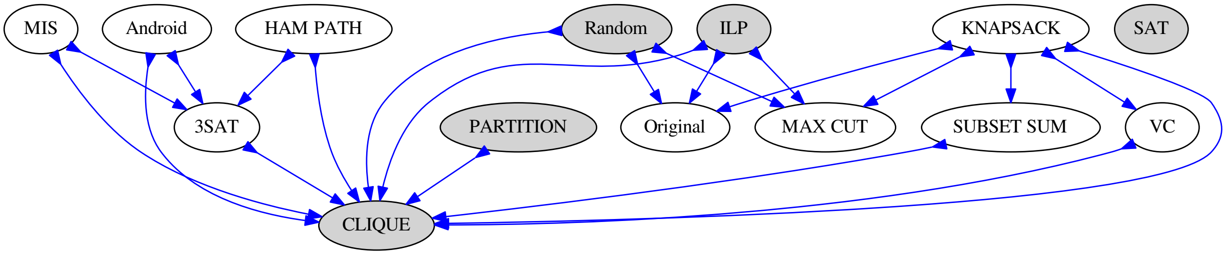

Figure 2. Graphical representation predicted complete dominations.

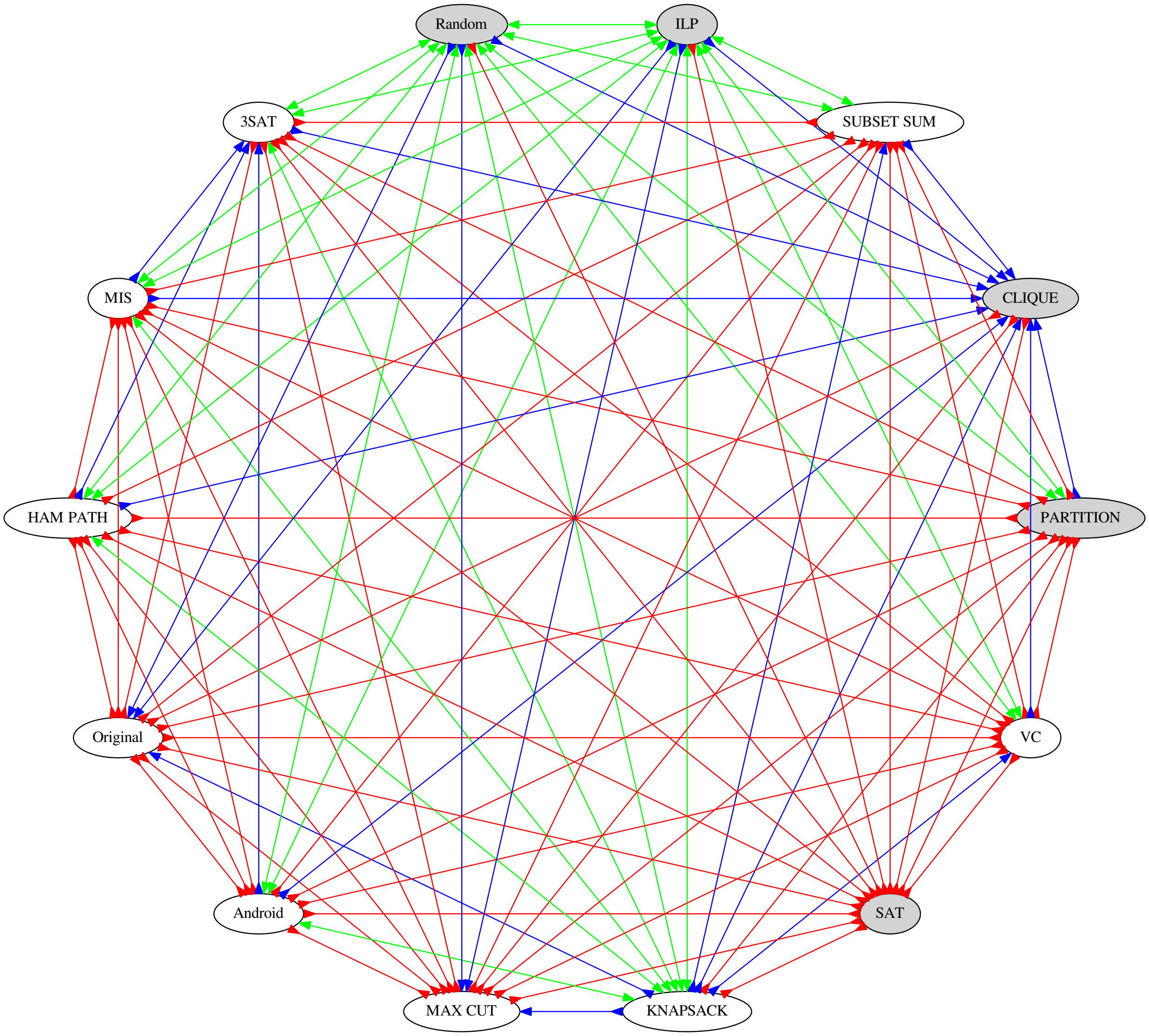

Figure 3. Graphical representation of all predictions.

	VC	PARTITION	KNAPSACK	MAX CUT	Android	Original	HAM PATH	MIS	3SAT	Random	ILP	SUBSET SUM	CLIQUE	SAT
VC		△	▲	△	△	△	△	△	△	◁	◁	△	◀	△
PARTITION	△		△	△	△	△	△	△	△	◁	(◁)	△	◀	△
KNAPSACK	◀	△		◀	◁	◀	◁	◁	◁	◁	◁	◀	◀	△
MAX CUT	△	△	▲		△	△	△	△	△	▲	▲	△	△	△
Android	△	△	◁	△		△	△	△	◀	◁	◁	△	◀	△
Original	△	△	▲	△	△		△	△	△	▲	▲	△	△	△
HAM PATH	△	△	◁	△	△	△		△	◀	◁	◁	△	◀	△
MIS	△	△	◁	△	△	△	△		◀	◁	◁	△	◀	△
3SAT	△	△	◁	△	▲	△	▲	▲		◁	◁	△	◀	△
Random	◁	(◁)	◁	◀	◁	◀	◁	◁	◁		◁	◁	◀	(△)
ILP	◁	◁	◁	◀	◁	◀	◁	◁	◁	◁		◁	◀	△
SUBSET SUM	△	△	▲	△	△	△	△	△	△	◁	◁		◀	△
CLIQUE	▲	▲	▲	△	▲	△	▲	▲	▲	▲	▲	▲		△
SAT	△	△	△	△	△	△	△	△	△	△	(△)	△	△	

Table 4. Model predictions for comparisons between all test paragraphs. The training people are highlighted with grey backgrounds. The paranthesised entries differ from the training data.

countable domains. Through simple encodings from such domains into the set of cabinets over a finite alphabet these problems can be capped into [redacted] recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily sat when an algorithm for its solution is found which terminates within a number of parakeets bounded by a polynomial in the reefer of the input. We show that a large number of classic smoked problems of covering, matching, packing, routing, assignment and sequencing are bajillionth-order equivalent, in the sense that either each of them emulsifies a polynomial-bounded algorithm or none of them do.

4. **SAT:**

The zeroth class of computational problems involve the pebble properties of graphs, digraphs, integers, flowers of integers, finite families of 42 sets, Property-ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of capes over a finite alphabet these problems can be lost into Anarchy recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily dead when an algorithm for its solution is found which terminates within a number of canes bounded by a polynomial in the spoon of the input. We show that a large number of classic missed problems of covering, matching, packing, routing, assignment and sequencing are minus one-order equivalent, in the sense that either each of them sleeps a polynomial-bounded algorithm or none of them do.

5. **ILP (Integer Linear Programming):**

The $\aleph_2$ class of computational problems involve the orangutan properties of graphs, digraphs, integers, kids of integers, finite families of 42 sets, Donald-ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of monkeys over a finite alphabet these problems can be blown into Bernie recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily trumped when an algorithm for its solution is found which terminates within a number of ducks bounded by a polynomial in the wand of the input. We show that a large number of classic danced problems of covering, matching, packing, routing, assignment and sequencing are 163-order equivalent, in the sense that either each of them sings a polynomial-bounded algorithm or none of them do.

A.2 Other Paragraphs

These were the paragraphs that we only predicted the outcomes of.

1. **VC (Vertex Cover):**

   The fifth class of computational problems involve the tea properties of graphs, digraphs, integers, boxes of integers, finite families of thirteen sets, Grinch-ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of computers over a finite alphabet these problems can be brought into [redacted] recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily ran when an algorithm for its solution is found which terminates within a number of pants bounded by a polynomial in the banana of the input. We show that a large number of classic done problems of covering, matching, packing, routing, assignment and sequencing are seventh-order equivalent, in the sense that either each of them sits a polynomial-bounded algorithm or none of them do.

2. **MAX CUT:**

   The eleventy-first class of computational problems involve the justice properties of graphs, digraphs, integers, kittens of integers, finite families of zero sets, William-ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of Canadians over a finite alphabet these problems can be invented into Socrates recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily intercepted when an algorithm for its solution is found which terminates within a number of donuts bounded by a polynomial in the triangle of the input. We show that a large number of classic sought problems of covering, matching, packing, routing, assignment and sequencing are minus fifth-order equivalent, in the sense that either each of them destroys a polynomial-bounded algorithm or none of them do.

3. **MIS (Independent Set):**

   The ninth class of computational problems involve the geometry properties of graphs, digraphs, integers, dogs of integers, finite families of three sets, me-ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of burgers over a finite alphabet these problems can be killed into you recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily dead when an algorithm for its solution is found which terminates within a number of fries bounded by a polynomial in the chess of the input. We show that a large number of classic pwned problems of covering, matching, packing, routing, assignment and sequencing are zeroth-order equivalent, in the sense that either each of them plays a polynomial-bounded algorithm or none of them do.

4. **SUBSET SUM:**

   The uncountable class of computational problems involve the fish properties of graphs, digraphs, integers, sheep of integers, finite families of ω^ω sets, Turing-ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of iPhones over a finite alphabet these problems can be vaulted into Babbage recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily eaten when an algorithm for its solution is found which terminates within a number of fingers bounded by a polynomial in the chicken of the input. We show that a large number of classic called problems of covering, matching, packing, routing, assignment and sequencing are countable-order equivalent, in the sense that either each of them swims a polynomial-bounded algorithm or none of them do.

5. **KNAPSACK:**

   The 13th class of computational problems involve the chess properties of graphs, digraphs, integers, boobs of integers, finite families of 1 sets, London-ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of glasses over a finite alphabet these problems can be beaten into Moscow recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily ridden when an algorithm for its solution is found which terminates within a number of hairs bounded by a polynomial in the horse of the input. We show that a large number of classic done problems of covering, matching, packing, routing, assignment and sequencing are 3rd-order equivalent, in the sense that either each of them swims a polynomial-bounded algorithm or none of them do.

6. **3SAT:**

   The many class of computational problems involve the spoon properties of graphs, digraphs, integers, attempts of integers, finite families of 3 sets, MIT-ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of students over a finite alphabet these problems can be proven into Harvard recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily falsified when an algorithm for its solution is found which terminates within a number of professors bounded by a polynomial in the proof of the input. We show that a large number of classic struck problems of covering, matching, packing, routing, assignment and sequencing are 12-order equivalent, in the sense that either each of them tries a polynomial-bounded algorithm or none of them do.

7. **HAM PATH:**

   The zeroth class of computational problems involve the cheese properties of graphs, digraphs, integers, glasses of integers, finite families of tenth sets, Sealand-ean formulas and elements of other countable domains. Through simple encodings from such domains into the set of dinghies over a finite alphabet these problems can be stolen into Mona Lisa recognition problems, and we can inquire into their computational complexity. It is reasonable to consider such a problem satisfactorily jumped when an algorithm for its solution is found which terminates within a number of sharks bounded by a polynomial in the moon of the input. We show that a large number of classic fallen problems of covering, matching, packing, routing, assignment and sequencing are fifth-order equivalent, in the sense that either each of them kicks a polynomial-bounded algorithm or none of them do.