Scientific Writing

January 10, 2025

About Writing Advice

Most of it sounds trivial.

About Writing Advice

The hard part is recognizing when it is needed.

About Writing Advice

There are no universal rules.

Try to implement the rule - then decide whether or not you want to use it.

Contents

Working/Writing Process

Structure of a Paper

Math in writing

Clarity

Language

Editing

In general

It’s okay if what you first write is a mess.

Editing is a thing.

Common Work Processes:

Agile – Discovery Writer, Pantser

- Write a bit
- Get feedback on it
- Implement feedback
- Repeat

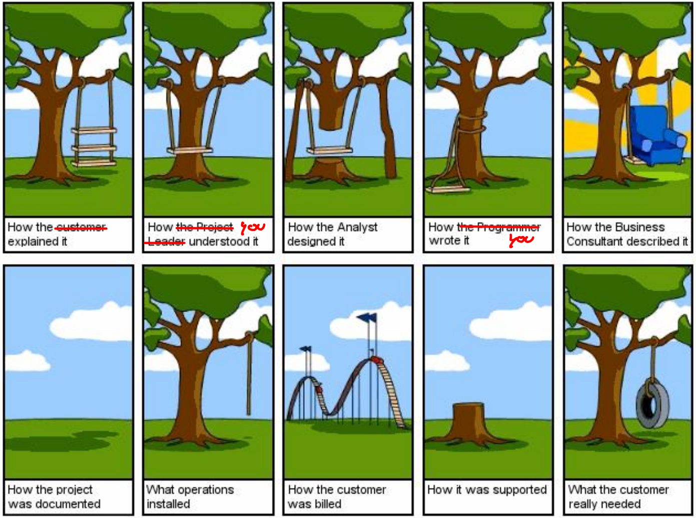
Waterfall Model – Plotter

- Plan out whole structure
- Write everything
- Get feedback
- Edit

Both processes are valid!

Talk to your advisor regularly!

advisor

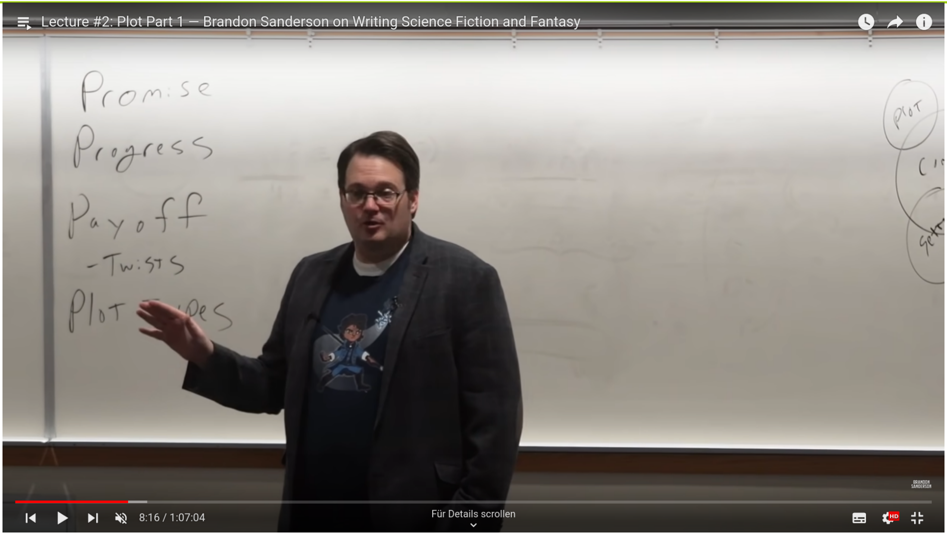

How the ~~customer~~ explained it
How ~~the Project~~ you ~~Leader~~ understood it
How the Analyst designed it
How ~~the Programmer~~ wrote it you
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

[https://effectivesoftwaredesign.com/2012/04/23/communication-problems-in-software-projects/]

General Structure

- Title
- Abstract
- Introduction and motivation
- Basic definitions
- Main part of your work
- Conclusion and further directions

Tell a story . . .



Lecture #2: Plot Part 1 — Brandon Sanderson on Writing Science Fiction and Fantasy

8:16 / 1:07:04

Für Details scrollen

Brandon Sanderson on Writing Science Fiction and Fantasy [San20]

Example: Harry Potter and the Chamber of Secrets

Promise

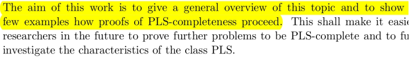
Progress

The Chamber of Secrets has been ope
enemies of the heir... be

Payoff

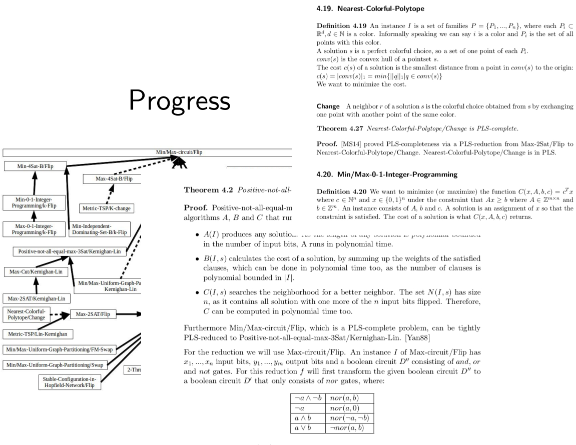
Example: Bachelor Thesis

Promise



The aim of this work is to give a general overview of this topic and to show few examples how proofs of PLS-completeness proceed. This shall make it easie researchers in the future to prove further problems to be PLS-complete and to fu investigate the characteristics of the class PLS.

Progress



Min/Max-circuit/Flip
Min-4Sat-B/Flip
Max-4Sat-B/Flip
Min-0-1-Integer-Programming/k-Flip
Metric-TSP/K-change
Max-0-1-Integer-Programming/k-Flip
Min-Independent-Dominating-Set-B/k-Flip
Positive-not-all-equal-max-3Sat/Kernighan-Lin
Max-Cut/Kernighan-Lin
Min/Max-Uniform-Graph-Pa Kernighan-Lin
Max-2SAT/Kernighan-Lin
Nearest-Colorful-Polytope/Change
Max-2SAT/Flip
Metric-TSP/Lin-Kernighan
Min/Max-Uniform-Graph-Partitioning/FM-Swap
Min/Max-Uniform-Graph-Partitioning/Swap
2-Thre
Stable-Configuration-in-Hopfield-Network/Flip

4.19. Nearest-Colorful-Polytope

Definition 4.19 An instance I is a set of families $P = \{P_1, ..., P_n\}$, where each $P_i \subset \mathbb{R}^d, d \in \mathbb{N}$ is a color. Informally speaking we can say i is a color and P_i is the set of all points with this color.
A solution s is a perfect colorful choice, so a set of one point of each P_i.
$conv(s)$ is the convex hull of a pointset s.
The cost $c(s)$ of a solution is the smallest distance from a point in $conv(s)$ to the origin:
$c(s) = |conv(s)|_1 = min\{||q||_1 | q \in conv(s)\}$
We want to minimize the cost.

Change A neighbor r of a solution s is the colorful choice obtained from s by exchanging one point with another point of the same color.

Theorem 4.27 *Nearest-Colorful-Polytope/Change is PLS-complete.*

Proof. [MS14] proved PLS-completeness via a PLS-reduction from Max-2Sat/Flip to Nearest-Colorful-Polytope/Change. Nearest-Colorful-Polytope/Change is in PLS.

4.20. Min/Max-0-1-Integer-Programming

Definition 4.20 We want to minimize (or maximize) the function $C(x, A, b, c) = c^T x$ where $c \in \mathbb{N}^n$ and $x \in \{0,1\}^n$ under the constraint that $Ax \geq b$ where $A \in \mathbb{Z}^{m \times n}$ and $b \in \mathbb{Z}^m$. An instance consists of A, b and c. A solution is an assignment of x so that the constraint is satisfied. The cost of a solution is what $C(x, A, b, c)$ returns.

Theorem 4.2 *Positive-not-all-*

Proof. Positive-not-all-equal-m algorithms A, B and C that rur

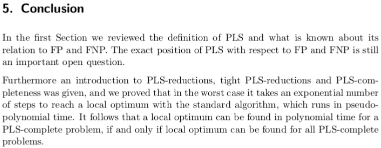
- $A(I)$ produces any solutio... As the length of any solution is polynomial bounded in the number of input bits, A runs in polynomial time.
- $B(I, s)$ calculates the cost of a solution, by summing up the weights of the satisfied clauses, which can be done in polynomial time too, as the number of clauses is polynomial bounded in $|I|$.
- $C(I, s)$ searches the neighborhood for a better neighbor. The set $N(I, s)$ has size n, as it contains all solution with one more of the n input bits flipped. Therefore, C can be computed in polynomial time too.

Furthermore Min/Max-circuit/Flip, which is a PLS-complete problem, can be tightly PLS-reduced to Positive-not-all-equal-max-3Sat/Kernighan-Lin. [Yan88]

For the reduction we will use Max-circuit/Flip. An instance I of Max-circuit/Flip has $x_1, ..., x_n$ input bits, $y_1, ..., y_m$ output bits and a boolean circuit D'' consisting of *and*, *or* and *not* gates. For this reduction f will first transform the given boolean circuit D'' to a boolean circuit D' that only consists of *nor* gates, where:

$\neg a \wedge \neg b$	$nor(a, b)$
$\neg a$	$nor(a, 0)$
$a \wedge b$	$nor(\neg a, \neg b)$
$a \vee b$	$\neg nor(a, b)$

Payoff



5. Conclusion

In the first Section we reviewed the definition of PLS and what is known about its relation to FP and FNP. The exact position of PLS with respect to FP and FNP is still an important open question.

Furthermore an introduction to PLS-reductions, tight PLS-reductions and PLS-completeness was given, and we proved that in the worst case it takes an exponential number of steps to reach a local optimum with the standard algorithm, which runs in pseudo-polynomial time. It follows that a local optimum can be found in polynomial time for a PLS-complete problem, if and only if local optimum can be found for all PLS-complete problems.

Promise

- Context and motivation.
- What is the hole that your work is filling?
- State what your result is.
- State what you are proving, implementing, testing.
- Make clear what will work and what won't.

⇒ The reader know what's going to happen.

Payoff

- What was proven or tested?
- What can be followed from the test results?
- Why is that meaningful?

Math in writing

- Don't start a sentence with a symbol.
- Don't use the symbols like $\Rightarrow, \forall, \exists$ in text, replace them by the corresponding words.
- Always define all your variables!
- State the type of a variable.
- Definitions of variables are written with ”$:=$“
- Symbols in different formulas should be separated by words
- Capitalize names like Theorem 1, Lemma 2, Algorithm 3, Method 4.

Example

$\forall (u, v)$, $W = W + w(u, v)$

$G = (V, E)$ is a graph. $\forall (u, v)$, $W = W + w(u, v)$

Let $G = (V, E)$ be a graph. $\forall (u, v)$, $W = W + w(u, v)$.

Let $G = (V, E)$ be a graph. For all (u, v), $W = W + w(u, v)$.

Let $G = (V, E)$ be a graph. For all edges $(u, v) \in E$, $W = W + w(u, v)$.

Let $G = (V, E)$ be a graph. For all edges $(u, v) \in E$, Algorithm 1 adds $W := W + w(u, v)$.

Let $G = (V, E)$ be a graph. For all edges $(u, v) \in E$, Algorithm 1 adds the weight of the edge $w(u, v)$ to the total sum of weights W, i.e., $W := W + w(u, v)$.

Math in writing

[My math tutor]:
Formulas and bullet points are part of the sentence. End them with a ”.“.

Bad:

Let $G = (V, E)$

- V is the set of vertices
- E is the set of edges

The function f computes the weight of all edges

$$f(E) := \sum_{e \in E} w(e)$$

Better:

Let $G = (V, E)$ be a graph where

- V is the set of vertices and
- E is the set of edges.

The function f computes the weight of all edges,

$$f(E) := \sum_{e \in E} w(e).$$

Math in writing

Even with formulas and symbols, it is still *text*.

Math in writing

[Mulzer]:
Don't use sources as noun. Use the name of the authors instead.

Bad: The complexity class UEOPL was introduced in 2018 by [Fea+18].

Better: The complexity class UEOPL was introduced in 2018 by Fearnly et al. [Fea+18].

Personal Tip: Always immediately add where you've got information from. Figuring out the sources at the end of writing is a pain.

Clarity

- Clarity is more important than avoiding repetition.
- Try to state things twice (e.g. as text and formula.)
- A picture says more than a thousand words.
- Short sentences are clearer than long sentences.
- Define *all* variables that are used.
- Consistency! Don't use the same variable name twice.
- Can that be misinterpreted somehow?
- Does there happen more in my head than on the page?

Avoid Vague Words

- it
- this
- they

Bad example:
The input of the algorithm is an edge of a graph. It is used to compute the shortest path.

What is referenced?

- any

Bad example: This property holds for any vertex.

This property holds for all vertices?
Thus property holds for some vertex?

How to avoid clarity issues?

- Let the text rest a while and read it again a week later.
- Let someone else read it.

  Let them just recount what they read.

General language-rules

- Don't use passive language.
- Don't use don't, hasn't, can't, . . .
- Pick either American or British English and stick with it.

Avoid Weasel Words

- Filler words
- Unnecessary for content
- Sentences are shorter when they are left out

- just
- a bit
- almost
- basically
- may
- quite
- relatively
- rather
- actually
- about
- even
- kind of
- somehow
- like
- ...

- bloß
- doch
- ein bisschen
- relativ
- nur
- irgendwie
- vielleicht
- ziemlich
- wahrscheinlich
- etwas
- sozusagen
- teilweise
- . . .

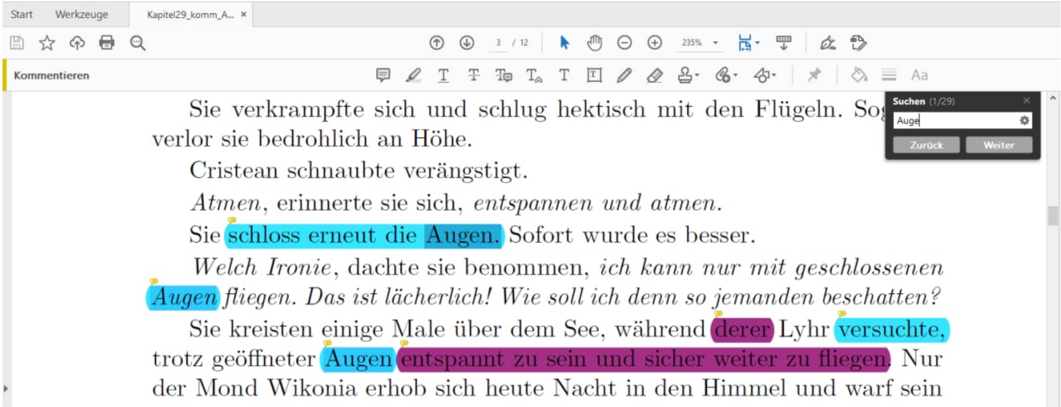
Avoid Weasel Words

Ich benutze **doch** keine Wieselwörter.
In Büchern gibt es sowas **doch** auch nicht.
Das merkt man **doch**, wenn man in jedem Satz ein unnötiges Wort hat.
So unaufmerksam kann **doch** niemand sein.
. . .
Doch.

Avoid Weasel Words

- If you still don’t believe me: Read someone elses work.

  Other peoples weasel words are much easier to spot.
- You don’t have to avoid them from the start.
- Know your tendencies.
- Do CTRL + F for your favorites before you hand in.



Sie verkrampfte sich und schlug hektisch mit den Flügeln. So verlor sie bedrohlich an Höhe.

Cristean schnaubte verängstigt.

Atmen, erinnerte sie sich, *entspannen und atmen.*

Sie schloss erneut die Augen. Sofort wurde es besser.

Welch Ironie, dachte sie benommen, *ich kann nur mit geschlossenen Augen fliegen. Das ist lächerlich! Wie soll ich denn so jemanden beschatten?*

Sie kreisten einige Male über dem See, während derer Lyhr versuchte, trotz geöffneter Augen entspannt zu sein und sicher weiter zu fliegen. Nur der Mond Wikonia erhob sich heute Nacht in den Himmel und warf sein

Would and had constructions

[Bis17]
Reduce unnecessary *would*, *had* and *have* constructions.
It's shorter.

To prove this, we have to create a successor function.
To prove this, we create a successor function.

About Writing Advice

Many things to keep in mind.

Editing

Now apply the advice.

Editing

1. Developmental edits
   - Content, correctness, structure
2. Line edits
   - Word choice, sentence structure, formulation
3. Copy edits
   - Are the line and page breaks nice?
   - Are pictures positioned nicely?

Tips for editing

- Track TODO's.
- Read your text aloud.
- Let someone else read your text.
- Read someone elses work.

Conclusion

Know your tendencies.

Don't panic.

In case of any doubts, talk to your advisor.

Bibliography

[Bis17] Shaelin Bishop. *18 Writing Hacks for Stronger Prose*. 2017. URL: https://www.youtube.com/watch?v=v45sfrLhLm4.

[San20] Brandon Sanderson. *Creative Writing Lecture at BYU*. 2020. URL: https://www.youtube.com/watch?v=jrIogch5DBU&list=PLSH_xM-KC3Zv-79sVZTTj-YA6IAqh8qeQ&index=2.

Further Resources

Ablauf einer Abschlussarbeit in der AG Software Engeneering:
(Kann als guideline natürlich auch auf andere Fachbereiche übertragen werden)
`https://www.mi.fu-berlin.de/w/SE/ThesisRules`

Lutz Prechelt über Wissenschaftliches Schrieben:
`https://www.mi.fu-berlin.de/inf/groups/ag-tech/links/prechelt99schreiben.pdf`

Notes on Lecture of Donald Knuth about Mathematical Writing:
`https://jmlr.csail.mit.edu/reviewing-papers/knuth_mathematical_writing.pdf`