What we talk about, when we talk about

Algorithms, Graphs, Tables, and Mathematics

Zhilei Ren

OSCAR Team, School of Software, Dalian University of Technology

March 28, 2018

Outline

Outline

About Me

Zhilei Ren

- Associate Professor
- In SSDUT since 2003
- `http://oscar-lab.org/people/~zren`
- `zren@dlut.edu.cn`
- Research interests:
  - Evolutionary computation
  - Data mining
  - Applications in software engineering

About OSCAR

The OSCAR Team

- Lead by Prof. He Jiang
- `http://oscar-lab.org`
- "**O**ptimizing **S**oftware by **C**omputation from **AR**tificial intelligence"
- or "**O**perating **S**ystem will **C**rash whenever my **A**lgorithm **R**uns"

About OSCAR

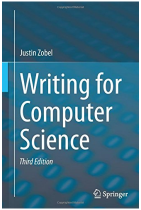
The OSCAR Team

- Lead by Prof. He Jiang
- `http://oscar-lab.org`
- "**O**ptimizing **S**oftware by **C**omputation from **AR**tificial intelligence"
- or "**O**perating **S**ystem will **C**rash whenever my **A**lgorithm **R**uns"

About this lecture

What will be covered

- Chapter 9–11 of **Writing for Computer Science**
- My experience on “where to find ideas” and “tools and tips”
- `https://git.oschina.net/zren/scientific-writing`



Outline

Algorithms

An algorithm must be seen to be believed.

– Donald Knuth
The Art of Computer Programming
vol.1: Fundamental Algorithms

Algorithms

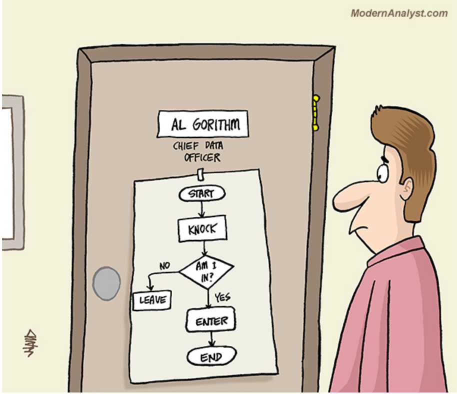


Figure 1: Flowchart for visiting Dr. Al Gorithm

Definitions of algorithm

- Informally, an algorithm is any **well-defined** computational procedure that takes some value, or set of values, as **input** and produces some value, or set of values, as **output** [11].
- In mathematics and computer science, an algorithm is a **self-contained sequence** of actions to be performed. Algorithms can perform **calculation**, **data processing** and **automated reasoning** tasks [2].

What to present in algorithms

The readers would expect to find

- The steps that make up the algorithm.
- The input and output, and the internal data structures used by the algorithm.
- The scope of application of the algorithm and its limitations.
- The properties that will allow demonstration of correctness, which might be formally expressed as pre- and post-conditions and loop invariants,
- A demonstration of correctness.
- A formal analysis of cost, for both space and time requirements.
- Experiments confirming the theoretical results.

Example of pseudocode

The **WeightedEdit** function computes the edit distance between two strings, assigning a higher penalty for errors closer to the front.

Input:	$S1, S2$: strings to be compared.
Output:	weighted edit distance
Variables:	$L1, L2$: string lengths
	$F[L1, L2]$: array of minimum distances
	W: current weighting
	M: maximum penalty
	C: current penalty

```
WeightedEdit(S1, S2):
L1 = len(S1)
L2 = len(S2)
M = 2 × (L1 + L2)
F[0,0] = 0
for i from 1 to L1
    F[i,0] = F[i − 1,0] + M − i
for j from 1 to L2
    F[0, j] = F[0, j − 1] + M − j
for i from 1 to L1
    C = M − i
    for j from 1 to L2
        C = C − 1
        F[i, j] = min(F[i − 1, j] + C,
                      F[i, j − 1] + C,
                      F[i − 1, j − 1] + C × isdiff(S1[i], S2[j]))
WeightedEdit = F[L1, L2]
```

Figure 2: Pseudocode meant for a machine, not an explanation for a reader

Example of pseudocode

WeightedEdit(s,t) compares two strings s and t, of lengths $k(s)$ and $k(t)$ respectively, to determine their edit distance—the minimum cost in insertions, deletions, and replacements required to convert one into the other. These costs are weighted so that errors near the start of the strings attract a higher penalty than errors near the end.

The major steps of the algorithm are as follows.

1. Set the penalty.
2. Initialize the data structure.
3. Compute the edit distance.

We now examine these steps in detail.

1. Set the penalty.

   The main property that we require of the penalty scheme is that costs reduce smoothly from start to end of string. As we will see, the algorithm proceeds by comparing each position i in s to each position j in t. Thus a diminishing penalty can be computed with the expression $p-i-j$, where p is the maximum penalty. By setting the penalty thus

   (a) Set $p \leftarrow 2 \times (k(s)+k(t))$

   the minimum penalty is $p-k(s)-k(t) = p/2$ and the next-smallest penalty is $p/2+1$. This means that two errors—regardless of position in the strings—will outweigh one.

2. Initialize data structures ...

Figure 3: The algorithm is explained and presented simultaneously. (Incomplete code)

Level of Detail

Algorithms should be specified in sufficient detail to allow them to be implemented without undue inventiveness.

5. (Matching.) For each pair of strings $s, t \in S$, find $N_{s,t}$, the maximum number of non-overlapping substrings that s and t have in common. ✗

Level of Detail

But don't provide too much detail. For example, loops are sometimes used unnecessarily in specification of algorithms.

3. (Summation.) $sum \leftarrow 0$. For each j, where $1 \geq j \geq n$,
   (a) $c \leftarrow 1$; the variable c is a temporary accumulator.
   (b) For each k, where $1 \leq k \leq m$, set $c \leftarrow c \times A_{jk}$.
   (c) $sum \leftarrow sum + c$. ✗

3. (Summation.) Set $sum \leftarrow \sum_{j=1}^{n}(\Pi_{k=1}^{n} A_{jk})$. ✓

Notation

Mathematical notation rather than programming notation

- Use "x_i" rather than "`x[i]`".
- Use "$\times$" or "$\cdot$" rather than "*" or "x".
- Avoid using constructs from specific programming, e.g., "$==$", "`a++`".
- Use mathematical notations, e.g., "$\sum$", "Π".

Environment of algorithms

Environment is important

- Specify the types of all variables, other than trivial items such as counters;
- Describe expected input and output, including assumptions about the correctness;
- State any limitations of the algorithm;
- Discuss possible errors that are not explicitly captured by the algorithm.
- State what the algorithm does.

Outline

Graphs

"And what is the use of a book", thought Alice, "without pictures or conversations?"

– Lewis Carroll
Alice in Wonderland

Graphs

Figure 4: Good results

Graphs

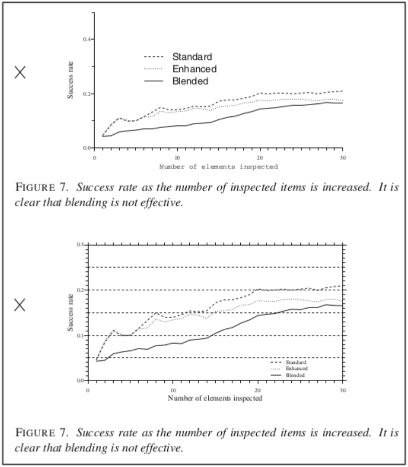


Figure 5: Badly designed graphs

Graphs

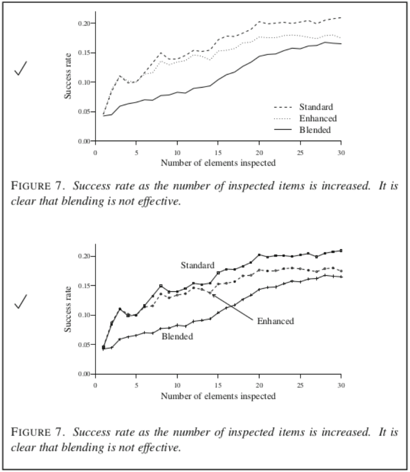


Figure 6: Graphs reconsidered

Graphs

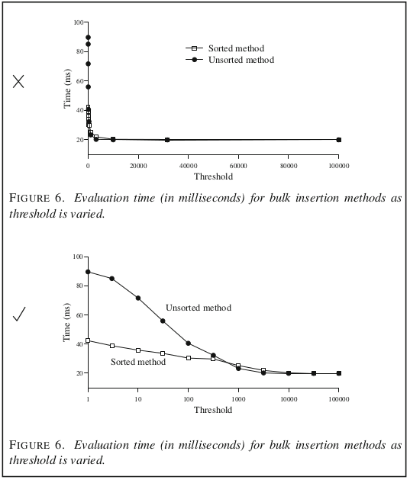


FIGURE 6. *Evaluation time (in milliseconds) for bulk insertion methods as threshold is varied.*

FIGURE 6. *Evaluation time (in milliseconds) for bulk insertion methods as threshold is varied.*

Figure 7: Choice of axis scaling

Graphs

✓

Data set	Method	
	A	B
Small, random	11.5	11.6
Large, random	27.9	17.1
Small, clustered	9.7	8.2
Large, clustered	24.0	13.5
All documents	49.4	60.1
First 1000	21.1	35.4
Last 1000	1.0	5.5

✓

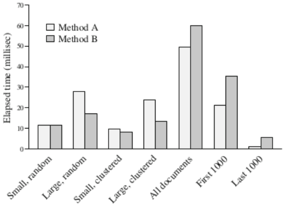


FIGURE 2. *Elapsed time (milliseconds) for methods A and B applied to data sets 1–7.*

Figure 8: Tables and Graphs focus on different aspect of data

Graphs

TABLE 8.4. *Tradeoff of space against time for methods A to G.*

×

Method	Space (%)	Time (ms)
A	1.0	7,564.5
B	31.7	895.6
C	44.7	458.4
D	97.8	71.8
E	158.1	18.9
F	173.7	1.4
G	300.0	0.9

✓

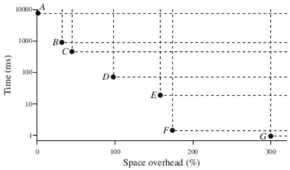


FIGURE 8.4. *Tradeoff of space against time for methods A to G. The boxed area to the right and above each point is of unacceptable performance: any method in that area will be less efficient with respect to both space and time than the point at the box's corner.*

Figure 9: Another table compared to a graph

Table

TABLE 6. *Statistics of text collections used in experiments.*

×

STATISTICS	SMALL	LARGE
Characters	18,621	1,231,109
Words	2,060	173,145
After stopping	1,200	98,234
Index size	1.31 Kb	109.0 Kb

TABLE 6. *Statistics of text collections used in experiments.*

✓

	Collection	
	Small	Large
File size (Kb)	18.2	1,202.3
Index size (Kb)	1.3	109.0
Number of words	2,060	173,145
— after stopping	1,200	98,234

Figure 10: Two versions of a table

Table

TABLE 2.1. *Impact on performance (processing time and effectiveness) of varying each of the three parameters in turn, for both data sets. Default parameter values are shown in parentheses. Note that $p =$ 100,000 is not meaningful for the data set* SINGLE.

Parameter	Data set			
	SINGLE		MULTIPLE	
	CPU (msec)	Effective (%)	CPU (ms)	Effective (%)
n ($k=10$, $p=100$)				
2	57.5	55.5	174.2	22.2
3	21.5	50.4	79.4	19.9
4	16.9	47.5	66.1	16.3
✓ k ($n=2$, $p=100$)				
10	57.5	55.5	174.2	22.2
100	60.0	56.1	163.1	21.3
1000	111.3	55.9	228.8	21.4
p ($n=2$, $k=10$)				
100	57.5	55.5	174.2	22.2
1000	13.8	12.6	19.8	2.1
10,000	84.5	56.0	126.4	6.3
100,000	—	—	290.7	21.9

Figure 11: Table with a deep hierarchy

Caption

×

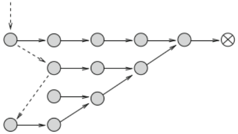


FIGURE 5. *Fan data structure.*

✓

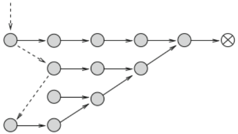


FIGURE 5. *Fan data structure, of lists with a common tail. The crossed node is a sentinel. Solid lines are within-list pointers. Dashed lines are inter-list pointers.*

Figure 12: Styles of caption

Outline

Mathematics

If you want to learn about nature, to appreciate nature, it is necessary to understand the language that she speaks in.

– Richard P. Feynman
The Relation of Mathematics to Physics

Mathematics

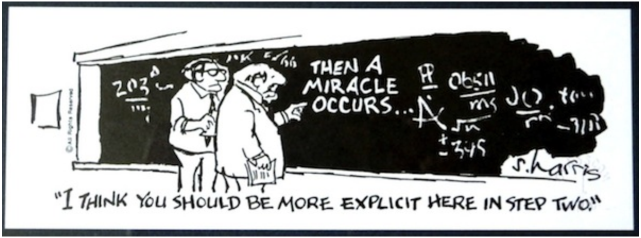


Figure 13: No magic in mathematics

Clarity is essential for mathematics

Clarity

- Mathematics gives solidity to abstract concepts.
- An inverted list for a given term is a sequence of pairs, where the first element in each pair is a document identifier and the second is the frequency of the term in the document to which the identifier corresponds. ✗
- An inverted list for a term t is a sequence of pairs of the form $\langle d, f \rangle$, where each d is a document identifier and f is the frequency of t in d. ✓

Mathematical terms can be confusing

Normal, usual, typical, strict, proper

- Normal: $f(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\sigma^2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$
- Usual and typical: Use these two words to avoid ambiguity.
- Definite (as in positive definite): In linear algebra, a symmetric $n \times n$ real matrix M is said to be positive definite if the scalar $z^{\mathrm{T}}Mz$ is positive for every non-zero column vector z of n real numbers.
- Strict: It is prudent to write "x is strictly positive" for $x > 0$ and "x is non-negative" for $x \geq 0$.
- Proper (as in proper subset): $A \subsetneq B$

Mathematical terms can be confusing (cont'd)

Any, average, mean, metric, measure

- Any: Avoid the word "any" in mathematical writing: sometimes it means "all" and sometimes it means "some".
- Average: is used loosely to mean typical. Only use it in the formal sense – of mean, that is, the arithmetic mean—if it is clear to the reader that the formal sense is intended. Otherwise use "mean" or even "arithmetic mean"
- Metric: When used in a formal context, a metric is expected to satisfy conditions such as the triangle inequality.
- Measure: also has a formal meaning, it is usually the less confusing of the two words, as it also has an appropriate informal usage. In mathematical contexts, use "measure" unless metric is intended.

Alphabets

Table 1: Symbols can be confused with each other

Symbol	Confused with
ϵ epsilon	*e*
ϵ epsilon	$\in$
η eta	*n*
ι iota	*i*
μ mu	*u*
ρ rho	*p*
τ tau	*t*
υ upsilon	*v*
ν nu	*v*
ω omega	*w*
$\vee$ or	*v*
$\cup$ union	*U*
$\propto$ proportional	α alpha
$\top$ top	*T*
$\emptyset$ empty set	0 zero
$\times$ times	*x*

Line breaks

Avoid letting a number, symbol, or abbreviation appear at the start of the line, particularly if it is the end of a sentence.

- ✗ We therefore have to make use of a further class variable, denoted by x. It allows ...
- ✓ We therefore have to make use of a further class variable, denoted by x. It allows ...
- ✗ The remaining values are irreducible, in which case it is clear that set $\mathcal{D}$ is not empty.
- ✗ Accesses to the new kind of file system typically require about 12 ms using our techniques.
- ✓ Accesses to the new kind of file system typically require about 12 ms using our techniques.
- ✗ In this case the problem can be simplified by using the term $f(x_1, \ldots,$ $x_n)$ as a descriptor.
- ✓ The problem is simplified if the term $f(x_1, \ldots, x_n)$ is used in this case as a descriptor.

Line breaks

Avoid letting a number, symbol, or abbreviation appear at the start of the line, particularly if it is the end of a sentence.

- ✗ We therefore have to make use of a further class variable, denoted by
  x. It allows ...
- ✓ We therefore have to make use of a further class variable, denoted by x.
  It allows ...
- ✗ The remaining values are irreducible, in which case it is clear that set
  $\mathcal{D}$ is not empty.
- ✗ Accesses to the new kind of file system typically require about 12
  ms using our techniques.
- ✓ Accesses to the new kind of file system typically require about 12 ms
  using our techniques.
- ✗ In this case the problem can be simplified by using the term $f(x_1, \ldots,$
  $x_n)$ as a descriptor.
- ✓ The problem is simplified if the term $f(x_1, \ldots, x_n)$ is used in this case
  as a descriptor.

Line breaks

Avoid letting a number, symbol, or abbreviation appear at the start of the line, particularly if it is the end of a sentence.

- ✗ We therefore have to make use of a further class variable, denoted by x. It allows ...
- ✓ We therefore have to make use of a further class variable, denoted by x. It allows ...
- ✗ The remaining values are irreducible, in which case it is clear that set $\mathcal{D}$ is not empty.
- ✗ Accesses to the new kind of file system typically require about 12 ms using our techniques.
- ✓ Accesses to the new kind of file system typically require about 12 ms using our techniques.
- ✗ In this case the problem can be simplified by using the term $f(x_1, \ldots,$ $x_n)$ as a descriptor.
- ✓ The problem is simplified if the term $f(x_1, \ldots, x_n)$ is used in this case as a descriptor.

Line breaks

Avoid letting a number, symbol, or abbreviation appear at the start of the line, particularly if it is the end of a sentence.

- ✗ We therefore have to make use of a further class variable, denoted by
  x. It allows . . .
- ✓ We therefore have to make use of a further class variable, denoted by x.
  It allows . . .
- ✗ The remaining values are irreducible, in which case it is clear that set
  $\mathcal{D}$ is not empty.
- ✗ Accesses to the new kind of file system typically require about 12
  ms using our techniques.
- ✓ Accesses to the new kind of file system typically require about 12 ms
  using our techniques.
- ✗ In this case the problem can be simplified by using the term $f(x_1, \ldots,$
  $x_n)$ as a descriptor.
- ✓ The problem is simplified if the term $f(x_1, \ldots, x_n)$ is used in this case
  as a descriptor.

Numbers

Digits or words?

- General writing guides recommend that large numbers should be written out in digits, such as 1,401 or 23, and that small numbers or round numbers should be spelt out.
- In technical writing, however, digits are generally preferred when quantities are being reported, and in particular when numbers are being compared.
- However, words are sometimes preferable, for approximate numbers and for numbers at the start of a sentence.
- Percentages should always be in figures.

Examples for numbers

- ✗ 1024 computers were linked into the ring.
- ✓ There were 1024 computers linked into the ring.
- ✗ The increase was over five per cent.
- ✓ The increase was over 5 per cent.
- ✓ The increase was over 5%.
- ✗ There were 14 512-Kb sets.
- ✓ There were fourteen 512-Kb sets.
- ✗ The sizes were 7.31 and 181 Kb, respectively.
- ✓ The sizes were 7.3 and 181.4 Kb, respectively.

Examples for numbers

- ✗ 1024 computers were linked into the ring.
- ✓ There were 1024 computers linked into the ring.
- ✗ The increase was over five per cent.
- ✓ The increase was over 5 per cent.
- ✓ The increase was over 5%.
- ✗ There were 14 512-Kb sets.
- ✓ There were fourteen 512-Kb sets.
- ✗ The sizes were 7.31 and 181 Kb, respectively.
- ✓ The sizes were 7.3 and 181.4 Kb, respectively.

Examples for numbers

- ✗ 1024 computers were linked into the ring.
- ✓ There were 1024 computers linked into the ring.
- ✗ The increase was over five per cent.
- ✓ The increase was over 5 per cent.
- ✓ The increase was over 5%.
- ✗ There were 14 512-Kb sets.
- ✓ There were fourteen 512-Kb sets.
- ✗ The sizes were 7.31 and 181 Kb, respectively.
- ✓ The sizes were 7.3 and 181.4 Kb, respectively.

Examples for numbers

- ✗ 1024 computers were linked into the ring.
- ✓ There were 1024 computers linked into the ring.
- ✗ The increase was over five per cent.
- ✓ The increase was over 5 per cent.
- ✓ The increase was over 5%.
- ✗ There were 14 512-Kb sets.
- ✓ There were fourteen 512-Kb sets.
- ✗ The sizes were 7.31 and 181 Kb, respectively.
- ✓ The sizes were 7.3 and 181.4 Kb, respectively.

Examples for numbers

- ✗ 1024 computers were linked into the ring.
- ✓ There were 1024 computers linked into the ring.
- ✗ The increase was over five per cent.
- ✓ The increase was over 5 per cent.
- ✓ The increase was over 5%.
- ✗ There were 14 512-Kb sets.
- ✓ There were fourteen 512-Kb sets.
- ✗ The sizes were 7.31 and 181 Kb, respectively.
- ✓ The sizes were 7.3 and 181.4 Kb, respectively.

Outline

Where to Find Ideas

A new idea comes suddenly and in a rather intuitive way. But intuition is nothing but the outcome of earlier intellectual experience.

– Albert Einstein

Where to Find Ideas

Figure 14: How to be creative

Where to Find Ideas

Sources of Idea

- Literature: critical reading and thinking
- Brainstorming: collective knowledge
- Tech news and documentations: my own experience

Where to Find Ideas

Critical Reading

The critical reading . . . implies a critical examination of the concepts used as well as of the soundness of the arguments and the value and relevance of the assumptions and the traditions on which the text is given [4].

Where to Find Ideas

Brainstorming

Brainstorming is a group creativity technique by which efforts are made to find a conclusion for a specific problem by gathering a list of ideas spontaneously contributed by its members [3].

Where to Find Ideas

Figure 15: A small example of idea from tech news

Outline

Tools and Tips

Man is a tool-making animal.

– Benjamin Franklin
Quoted by James Boswell
in The Life of Samuel Johnson

Tools and Tips

Figure 16: The first museum

Tools and Tips

Tools that make your research journey easier

- LaTeX[1]
- Visio
- Excel
- Origin Lab [7]
- R [10]
- Python [9] [a]
- Version Control System (VCS) (git [5], hg [6], etc)

[a] https://bitbucket.org/rezilla/open-lecture-notes.git

Recommending LaTeX

LaTeX source

```
\begin{eqnarry}
  E &=& mc^2
  \label{eq1}\\
  x &=& \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}
  \label{eq2}
\end{eqnarry}
Equations \ref{eq1} and \ref{eq2} indicate the mass-energy equation,
and the root of quadratic equations, respectively.
```

$$E = mc^2 \tag{1}$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{2}$$

Equations 1 and 2 indicate the mass-energy equation, and the root of quadratic equations, respectively.

Recommending LaTeX

LaTeX source

```
\begin{eqnarry}
  E &=& mc^2
  \label{eq1}\\
  x &=& \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}
  \label{eq2}
\end{eqnarry}
Equations \ref{eq1} and \ref{eq2} indicate the mass-energy equation,
and the root of quadratic equations, respectively.
```

$$E = mc^2 \tag{1}$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{2}$$

Equations 1 and 2 indicate the mass-energy equation, and the root of quadratic equations, respectively.

Microsoft Visio

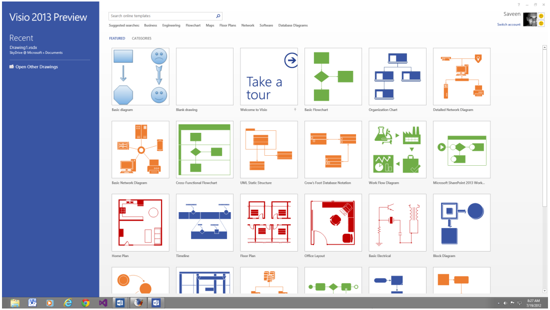


Figure 17: Visio

Microsoft Excel

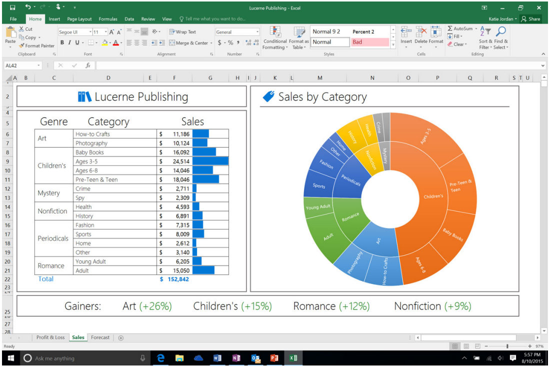


Figure 18: Excel

Origin Lab

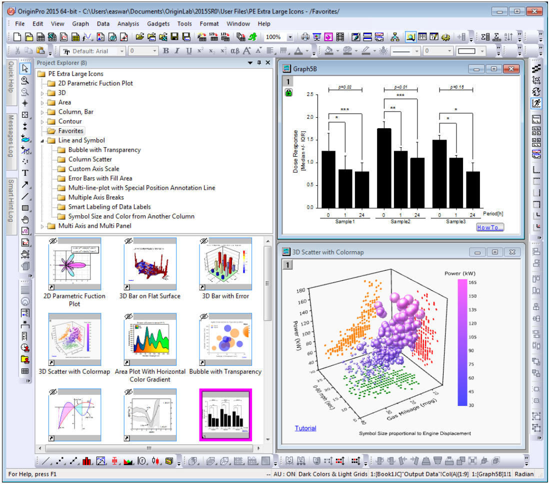


Figure 19: Orign Lab

Recommending R (ggplot2)

```
> library(ggplot2)
> p <- ggplot(data=diamonds, aes(x=carat, y=price, color=cut)) + geom_point()
> ggsave(filename="point.png", width=5, height=5, p)
```

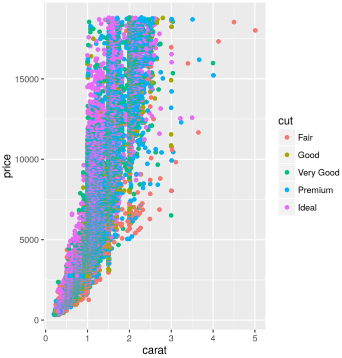


Figure 20: price vs. carat

Recommending R (xtable)

> library(xtable)
> xtable(diamonds[1:2,1:5], caption="The diamonds data.frame")

```
\begin{table}[ht]
\centering
\begin{tabular}{rrlllr}
  \hline
 & carat & cut & color & clarity & depth \\
  \hline
  1 & 0.23 & Ideal & E & SI2 & 61.50 \\
  2 & 0.21 & Premium & E & SI1 & 59.80 \\
   \hline
\end{tabular}
\caption{The diamonds data.frame} % move to the top manually
\end{table}
```

Table 2: The diamonds data.frame

	carat	cut	color	clarity	depth
1	0.23	Ideal	E	SI2	61.50
2	0.21	Premium	E	SI1	59.80

Recommending Python

```
#!/usr/bin/python3

import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(0, 10, 10000)
y1 = np.sin(x)
y2 = np.sqrt(x)
graph, axes = plt.subplots(2, 1)
for i in range(2):
    axes[i].set_xlabel("x")
    axes[i].grid()
axes[0].plot(x, y1)
axes[1].plot(x, y2)
axes[0].set_ylabel("sin(x)")
axes[1].set_ylabel("sqrt(x)")
graph.savefig("graph.png")
```

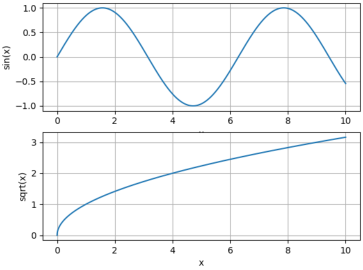


Figure 21: Example of matplotlib

Recommending Version Control System

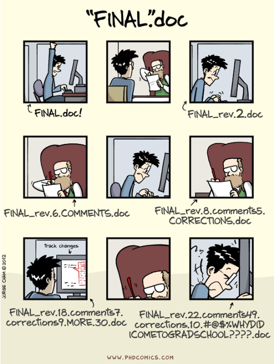


Figure 22: Why we need VCS

Tools and Tips

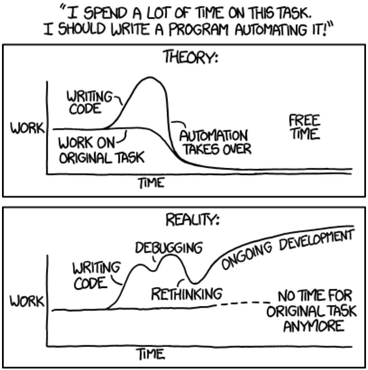


Figure 23: When the cure is worse than the disease...

Tools and Tips

Further reading

- The Ph.D. grind [12]
- You and your research [13]
- PHD comics [8]

References I

[1] LaTeX. http://www.latex-project.org/. Accessed: 2018-03-25.

[2] Algorithm. https://en.wikipedia.org/wiki/Algorithm. Accessed: 2018-03-25.

[3] Brainstorming. https://en.wikipedia.org/wiki/Brainstorming. Accessed: 2018-03-26.

[4] Critical reading. https://en.wikipedia.org/wiki/Critical_reading. Accessed: 2018-03-26.

[5] Git. https://git-scm.com/. Accessed: 2018-03-26.

[6] Mercurial. https://mercurial-scm.com/. Accessed: 2018-03-26.

[7] Origin Lab. http://originlab.com/. Accessed: 2018-03-26.

[8] Phd comics. http://phdcomics.com/. Accessed: 2018-03-26.

[9] Python. https://python.org/. Accessed: 2018-03-28.

[10] The R project for statistical computing. https://r-project.org. Accessed: 2018-03-26.

[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms (third edition)*. MIT press Cambridge, 2009.

References II

[12] P. Guo. The ph.d. grind.
http://pgbovine.net/PhD-memoir.htm.
Accessed: 2018-03-26.

[13] R. Hamming. You and your research.
http://www.cs.virginia.edu/~robins/YouAndYourResearch.html.
Accessed: 2018-03-26.

Thanks

Zhilei Ren
zren@dlut.edu.cn