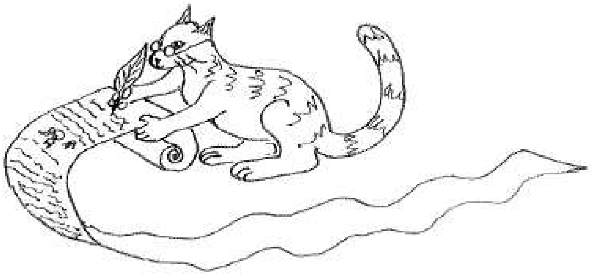
Scientific Writing (3 ects)

Wilhelmiina Hämäläinen

March 8, 2006



1. How to write scientific texts in computer science?
2. How to write in English?
3. How to write a master thesis?

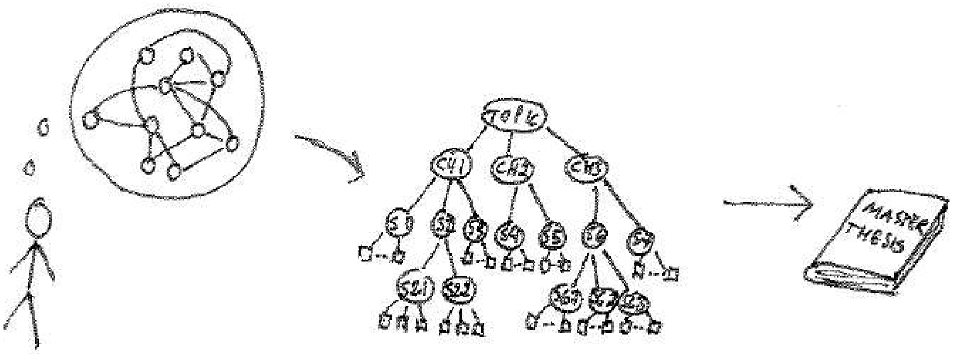
1 Goal 1: How to write scientific text is cs?

- general style
- how to use references
- equations, pictures, tables, algorithms
- useful tools (latex, bibtex, picture editors)

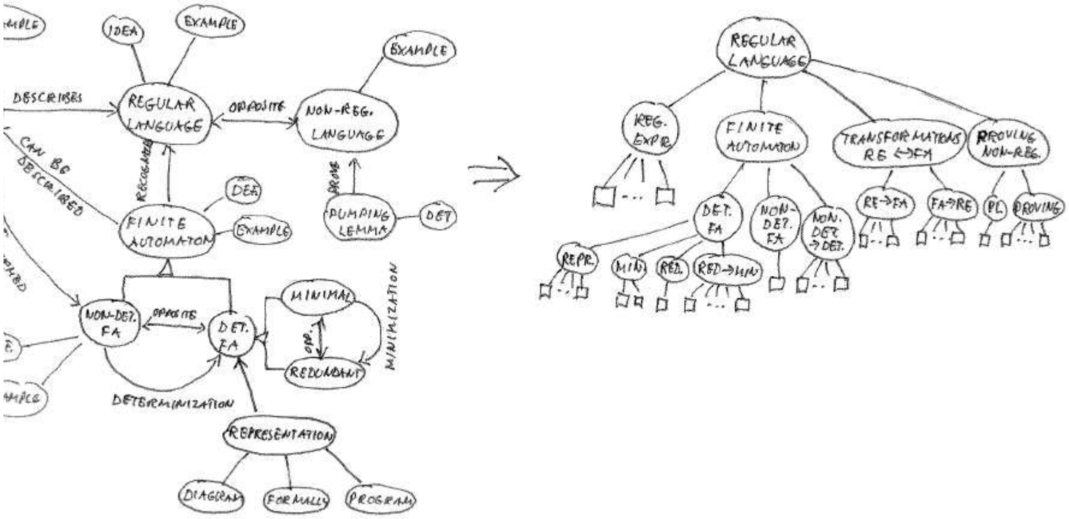
1.1 Problem

Writing w is a mapping from a set of ideas I to a set of scientific texts S, $w : I \rightarrow S$.

Problem: Given a set of ideas $i \in I$, produce $f(i) \in S$



1.2 Example



1.3 Instructions

1. Organize your ideas in a hierarchical manner, as a tree of ideas t (”minimal spanning tree” of idea graph)

2. Write the tree t as text such that

- The root node of t corresponds to your topic (title)
- Its children correspond to chapters
- Their children and grand-children correspond to sections and subsections
- Leaf nodes correspond to paragraphs (actual text)

1.4 Writing tree t

Each node $n \in t$ contains three fields:

- $title(n)$: the main title or the name of the chapter, section or subsection. In leaf nodes (paragraphs) $NULL$

- *children*(n): n's children (chapters, sections or subsections). In leaf node $NULL$.
- *content*(n): description of the idea in n. In non-leaf nodes very brief, in leaf nodes longer.

The following algorithm descibes how to walk through t in preorder and write it as a sequence $s \in S$ (scientific text):

Alg. 1 WriteTree(t)

```
Input: tree of ideas $t$
Output: scientific text $s$

begin
   Write $title(n)$
   if ($n$ is not leaf node)
      begin
      Writing an introductory paragraph:
         Write $content(n)$
         for all $u = child(n)$
            Write $title(u)$
         for all $u = child(n)$
            WriteTree($u$)
      end
   else
      Writing a main paragrap:
      Write $content(n)$
end
```

1.5 Properties of a good tree t

- t is balanced: all paths from the root to a leaf are approximately of equal length, usually ≤ 4 or at most ≤ 5.
- Each node in t has a reasonable number of children k: $k \leq 2$ and typically $k \leq 7$ (in maximum $k = 10$)
- For all leaf nodes n, the sizes of *content*(n) are balanced: each paragraph contains at least two sentences, but is not too long (e.g. ≤ 7 or ≤ 10 sentences)
- For all non-leaf nodes m, the sizes of *content*(m) are balanced. These introductory paragraphs can be very brief. They just give an overview what will be covered in that chapter or section. Exceptionally you can use more than one paragraph. Notice that it is possible to skip them totally, but be systematic!
- For all leaf nodes n_i in preorder, *content*(n_i) can refer only to previously written contents *content*(n_1),...,*content*(n_{i-1}). E.g. you cannot define deterministic automaton as an opposite of non-deterministic automaton, if you haven't given the definition of non-deterministic automaton, yet. Exception: you can briefly advertise what will be described in the future. E.g. "This problem is solved in Chapter X".

2 Goal 2: How to write English?

Every week we will spend some time with English grammar and expressions.

We will practice at least the following topics:

- dividing the text into paragraphs, sentences and clauses
- possessive case (expressing the owner)
- verb tense and number
- word order in sentences
- use of articles
- punctuation
- useful words and expressions

Other important topics??

Idea: personally selected exercises!

3 Goal 3: How to write a master thesis?

Writing a master thesis is not just writing, but you have to read a lot of material, make experiments, and analyze the results.

The process has the same phases as a software project or any problem solving activity:

1. **Defining the problem**: Discuss with your supervisor and define what is the problem. Try to understand it in a larger context: other related problems and subproblems. Read some introductory article about the topic or select the main books written about your topic. You can already generate several ideas how to solve it, but don't fix anything, yet.
2. **Specification**: Specify your topic carefully. Don't take too large topic! Invent a preliminary title for your thesis and define the content in a coarse level (main chapters). Ask your supervisor's approval! Decide with your supervisor what material you should read or what experiments to make.
3. **Design**: Define the content more carefully: all sections and a brief description what you will write in each of them. Define the main concepts you will need and fix the notations. Then you can write the chapters in any order you want. Make also a work plan: what you will do and when.
4. **Implementation**: You can write the thesis after you have read all material or made all experiments. However, you can begin to write some parts already when you are working. Often you have to change your design plan, but it is just life! Ask feedback from your supervisor, when your work proceeds.
5. **Final work**: Check language and spelling, missing or incomplete references. Check that the structure is coherent. Write an abstract.

Note: In practice it is easier to write other chapters, if you have an introduction, which defines the problem. However, often you have to write the introduction again in the end, when everything else is ready. Conclusions are also written in the end.

4 Course performance

- Participating contact teaching (obligatory)
- Solving weekly exercises (in computer classes and some homework)
- Writing a summary (about 1 page) from one article
- Writing a larger paper (where the summary is utilized)

  - 10-15 pages
  - based on 5-10 articles

- Extra points for small extra tasks!

4.1 How the course is evaluated??

Only the final paper? Or

- Summary 0–10%?
- Larger paper 60–90%?
- Exercises 0–10%?
- Course activity? Personal development?

Suggestion: in evaluation we take into account how much the student has imporved compared to her/his starting point. I.e. not only the results, but also the process is evaluated.

4.2 Extra tasks

The teacher will invent the tasks during the course. The idea is to test some tool and write brief instructions for other students how to use it. In addition, you can demonstrate the tool in computer classes. The tasks can be solved in pairs.
E.g. a graphical editor for latex, good (vector-graphic) drawing tools for windows.

4.3 Other notes

- All writing tasks must be written by latex!
- Make your own course course page and collect all material there (e.g. personal learning goals, vocabulary lists, task solutions, etc.)
  → the result is your ”learning portfolio”, which helps you to focus your learning and helps your teacher to give you individual feedback and supervison.

5 Topics and first articles

Each article is about 10-20 pages.

1. Ensamble learning (model averaging)
   Valentini, G. and Masulli, F.: Ensembles of learning machines

2. Empirical comparison of classification methods
   Quinlan, J.R.: Comparing connectionist and symbolic learning methods.

3. Automatic construction of concept maps
   Canas, A.J., Carff, R. et al.: Concept maps: integrating knowledge and information visualization.

4. Computerized adaptive testing
   Desmarais, M.C.: A Bayesian student model without hidden nodes and its comparison with item response theory. **You can read only the first 19 pages, and skip section ”Experimental evaluation of the approaches”**

5. Feature extraction by principal component analysis (PCA) and independent component analysis (ICA)
   Liu, C. and Wechsler, H.: Comparative assessment of independent component analysis (ICA) for face recognition **and** Norris, A.: Multivariate analysis and reverse engineering of signal transduction pathways. **Only Section 2.2**.

6. Mining temporal data (episodes)
   Laird, P.: Identifying and using patterns in sequential data. **or** Mannila, H., Toivonen, H. and Verkamo, A.I.: Discovering frequent episodes in sequences. Data mining and knowledge discovery 1, 259-289, 1997

7. Transfinite Turing machines
   Hamkins, J.D.: Infinite time Turing machines.

8. Probabilistic clustering
   Fraley, C. and Raftery, A.E.: Model-based clustering, discriminant analysis, and density estimation. **You can skip Sections 6-9**

9. Automatic assessment of students' program codes
   Sison, R.C., Numao, M. and Shimura, M.: Multistrategy discovery and detection of novice programmer errors. Machine Learning 38, 157-180, 2000.

10. Social filtering (collaborative learning)
    Shardanand, U. and Maes, P.: Social infromation filtering: algorithms for automating 'word of mouth'.

11. Bayesian student models in intelligent tutoring systems
    Martin, J. and Vanlehn, K.: Student assessment using Bayesian nets.

12. P versus NP question
    Sipser, M.: The history and status of the P versus NP question.

13. Bloom filters
    Broder, A. and Mitzenmacher, M.: Network Applications of Bloom Filters: A Survey.

14. Bit-parallel string matching
    Baeza-Yates, R. and Gonnet, G.H.: A new approach to text searching. Communications of the ACM, 35:74–82, Oct 1992.

6 Enlarging your vocabulary

Task: Read the given text extracts and underline all words and expressions which you don't know. How many of them you understand in the given context?

6.1 Hints

1. The best way is to read a lot of English texts! You learn also the use of terms and expressions in the right context.

2. Collect your own vocabulary where you list useful words and experssions you want to learn. You can divide the words into two categories: general words and expressions and computer science words and expressions.

3. It is easier to remember new words when you invent your own (maybe funny) sentences where you use them. E.g. if you want to learn expressions "suffer from" and "generalize beyond":
   "*Dogsifier* method suffers from overfitting, while *Catsifier* can easily generalize beyond the training set."

4. When you use a new word from dictionary, check carefully how it is used. English-to-English dictionaries are often useful.

Other hints?

7 Dividing the text into paragraphs, sentences, and clauses

Task 1: (In pairs) Read the given glossary
`http://orwell.ru/library/others/style/e/eap_01.htm`
of grammatic terms and check that you understand the meaning of the following terms:

adjective, adverb, articles, clause, compound sentence, dependent clause, independent clause, infinite, main clause, noun, number, object, phrase, predicate, preporsition, relative caluse, relative pronoun, subject, subordinate clause, verb, voice.

If some terms are unfamiliar, discuss with your partner, if s/he can explain them. Ask also if somebody knows the corresponding term in your mother tonque.

Basic rule: a paragraph has only one topic, a sentence expresses only one idea, a clause describes an atomic proposition.

Task 2: Now we will analyze your own writings. Change texts with your pair and analyze the division to paragraphs, sentences, and clauses in it. Mark into text

- When two paragraphs should be combined or one paragraph divided.
- If paragraphs are uncoherent (i.e. they cover several diverse topics)
- When two sentences should be combined or one should be divided.