Scholarship Skills

Andrew Black & David Maier
Portland State University

Structure of a Paper

Structuring a Paper

As with previous lectures, some of this material is from David Maier's class notes, and much is from Sandra Oster's seminar notes from circa 1985–1987.

Later additions by Tim Sheard and Andrew Black

Applicability

This material applies to essentially all papers

long, short, middle-sized, technical, survey, position …

We will talk later about the specifics of Journal Articles, Conference papers, and Dissertations

The Title

Make title precise

Some problems on graphs.

better:

Finding Hamiltonian circuits in directed graphs.

better still:

Parallel algorithms for finding Hamiltonian circuits in directed graphs.

State your result, if you have one

A complexity result for coding.

better:

Maximal prefix compression is NP-complete.

Use an *action verb* if you can.

Techniques for agent implementation.

better:

How to implement agents in Java.

Sometimes a witty title can be effective.

Nineteen dubious ways to compute the exponential of a matrix.

But avoid clichés

Agents considered harmful.

Author

Pick a professional name and stick with it.

- David Maier
- Dave Maier
- D. E. Maier
- Check with co-authors about preferences.

Order of authors

- alphabetical is common in CS when contribution is roughly equal
- Sometimes authors are grouped, then alphabetical within each group
- discuss it when you start the paper — or group of papers.

Organization and e-mail address are useful

- may be required — or prohibited!

Abstract

Abstract should be a "mini-paper": state motivation, problem, approach, and results

No citations — exception is abstracts for publication in conference programs.

Don't make the abstract a table of contents, or a condensation of the introduction

Repeating phrases in the introduction is annoying

Be specific

Not: "we consider three problems" — say what the problems are!

Do give away the ending—this is not a teaser for a mystery novel!

Abstract explains the whole paper.
Readers "shop" by looking at abstracts.

Try not to start with "In this paper" or "This paper presents"

Usually the last thing to write … and the first thing to write.

Maier on Abstracts

Each section of paper is represented by at least one sentence in the abstract. (Hence the abstract gets written last.)

Tell the reader about the important results!

Beck on Abstracts

The abstract is your four-sentence summary of the conclusions of your paper. Its primary purpose is to get your paper into the A pile. Most PC members sort their papers into an A pile and a B pile by reading the abstracts. The A pile papers get smiling interest; the B pile papers are a chore to be slogged through. By keeping your abstract short and clear, you greatly enhance your chances of being in the A pile.

Kent Beck, Panel session at OOPSLA '93

http://www.acm.org/sigplan/oopsla/oopsla96/how93.html

"I try to have four sentences in my abstract. The first states the problem. The second states why the problem is a problem. The third is my startling sentence. The fourth states the implication of my startling sentence. An abstract for this paper done in this style would be:

> The rejection rate for OOPSLA papers is near 90%. Most papers are rejected not because of a lack of good ideas, but because they are poorly structured. Following four simple steps in writing a paper will dramatically increase your chances of acceptance. If everyone followed these steps, the amount of communication in the object community would increase, improving the rate of progress.

"Well, I'm not sure that's a great abstract, but you get the idea.

"I always feel funny writing an abstract this way. The idea I thought was so wonderful when I started writing the paper looks naked and alone sitting there with no support. *I resist the temptation to argue for my conclusion in the abstract*. I think it gives the reader more incentive to carefully read the rest of the paper. They want to find out how in the world you could possibly say such an outrageous thing.

Example Abstract

Solutions abound for pretty-printing program text. Almost all work to date concerns how to write pretty-printers. We concentrate instead on what to print. We evaluated the output of sixteen different programming tools. Programmers gave subjective ratings and were also measured for speed and accuracy on understanding and modification tasks. We found that positive subjective ratings correlate well with performance on program understanding, but not with performance on maintenance. Surprisingly, formats with the least white space gave the most accuracy on both kinds of tasks.

Introduction

- The introduction should typically be accessible to a wider audience than the whole paper. Should interest, not bore, the reader.
- Often written next-to-last.
- *Don't* simply repeat the abstract, or even quote verbatim from it.
- Literature review can live in the introduction.
- Avoid introducing lots of notation and definitions in the introduction
  - If necessary, add a "notation" section after
- Fit your work into the larger context of your field.

Introduction

Tells the reader what problem you are solving

A good first sentence is essential

> Functional programmers are fond of producing elegant algorithms for pretty-printing program text; unfortunately, few of those algorithms produce elegant output.

Often ends with a summary of chapters or sections (the reading preparation).

Black almost never reads this!

Instead, use this space to tell me how you are going to tell the story.

or, list the contributions, and mention in passing which section expounds on each.

Is a section missing? Should that section be in the paper?

Example: Cloud Haskel

The contributions of this paper are as follows.

- A description of Cloud Haskell's interface functions (Sections 2 and 3). Following Erlang, our language provides a system for exchanging messages between lightweight concurrent processes, regardless of whether they are running on one computer or on many. We also provide functions for starting new remote processes, and for fault tolerance, which closely follow Erlang. However, unlike Erlang, Cloud Haskell also allows shared-memory concurrency *within* one of its processes.

- An additional message-passing interface that uses multiple *typed channels* in place of Erlang's single untyped channel (Section 4). Each channel is realized as a pair of ports; while the send port can be transmitted over the network, the receive port cannot.

Example: Cloud Haskel

- An additional message-passing interface that uses multiple *typed channels* in place of Erlang's single untyped channel (Section 4). Each channel is realized as a pair of ports; while the send port can be transmitted over the network, the receive port cannot.

- A method for serializing function closures that enables higher-order functions to be used in a distributed environment (Section 5). Starting a remote process means sending a representation of a function and its environment across the network; our approach makes the environment *explicit*, and thus gives the programmer control over the cost of the message.

- A demonstration of the effectiveness of our approach in the form of an implementation (discussed in Sections 6 and 7, and a complete example application (Section 8). We also provide performance measurements from the k-means clustering algorithm, an iterative algorithm for partitioning data points into natural groups (Section 9).

Typical structure of an introduction:

1. Problem Statement
   - Situation
   - Problem sentence(s)
     - *State the problem early!*
   - Motivation and Background
   - Technical issues, approach
   - Results

2. Reading Preparation
   - Communicative Purpose
   - Forecast Statement

Introduction

Problem Statement

Situation — introduces main topic and reviews past literature about the topic.

This review points out significant features of others' findings.

(Not exhaustive review. Not evaluative.)

Includes key definitions.

The Markov Decision Process (MDP) framework introduced by Bellman [1] is a good way of mathematically formalizing a large class of sequential decision problems involving an agent that is interacting with an environment. Generally, an MDP is defined in such a way that the agent has complete knowledge of the underlying state of the environment.

Problem Statement (continued)

Problem Sentence — explains the main issue you're addressing. Should be concise. Points to the main problem(s) in previous approaches that you're addressing.

> To date, the best known exact algorithms to solve POMDPs have trouble coping with problems containing only a few dozen states [5].

Technical Issues — Specific topics, investigative purpose, questions, approaches, more literature review.

> This is the "meat" of the introduction: it gives your purpose, direction, and approach.

Results — brief summary of your important results and how they contrast with previous work.

The entire Problem Statement tells the reader:

- *What* the problem is that you solve
- *Why* the problem is important
- *Where* the problem arises
- The *benefits* and *characteristics* of a good solution (efficiency, accuracy, robustness, theoretical understanding)
- The *current status* of the problem
- The *form* previous solutions take (literature review)
- The *approach* you're taking, and how it's motivated
- What the new approach *accomplishes*

Reading Preparation — tells reader the "lay of the land" of the rest of the paper.

Communicative Purpose — explains what the reader can expect the article to accomplish. Identifies tasks as a *writer*, not as a researcher. (Tasks as a researcher are identified in the Technical Issues section.)

cap ~~In this paper~~ we present empirical results comparing three approaches to belief-state reinforcement learning. The most direct approach is the use of a ...

Forecast Statement — Lists the topics that the paper discusses, in the order in which they are presented in the paper.

were

cap ~~We report on~~ the following visual functions measured with a 1.5 degree radius of the foveola: (a) blue-cone mediated sensitivity, (b) sensitivity at absolute threshold ...

Literature Review

Provides background information, situates your contribution in the context of previous work.

Suggests your level of familiarity with discipline.

Allows you to draw attention to work you believe is central and folks should read.

Provides basic definitions and vocabulary.

Can be:

- Topical, or
- Chronological

Literature Review

Be careful about statements like

~~However~~ to the best of our knowledge, there have been no investigations on the effect of different ...

How much literature review?

Depends on length of paper — introduction usually not more than 1/4 of total paper length.

Depends on scope of investigation (need to provide background for all topics you investigate).

Depends on audience's knowledge.

Depends on tone you want to set — e.g., introducing methods or points of view from another discipline.

Literature Review

Literature review can also come *after* the introduction — *if* the problem and the contributions can be understood without it.

Literature Review — Related Work

What constitutes relatedness?

similar problem

similar approach

alternative investigation

Characterize work of others well-enough that you can distinguish it from your own.

A list of sources is rarely adequate

You have to *explain the relationship* to your work.

Methodology

Explains *what* you did.

- Objectives of the process.
- Techniques.
- Details of procedures.
- Data analysis.

Reader should be able to judge the validity of your results.

Best outcome: a well-versed reader can reproduce your results. (More challenging for experimental than for theory papers.)

Organize into sections and subsections.

Put key ideas in lead position.

Ways to Organize a paper

The form "Problem–Solution–Defense" works for 80% of papers

Experimental

- Problem: distinguishing phonemes
- Solution: NN classification based on frequency features
- Defense: Precision measurements from an experiment

Theoretical

- Problem: need for increased concurrency
- Solution: new locking protocol
- Defense: Correctness proof, comparison to other methods

Other Examples

Algorithms

Problem: 2-D pattern matching

Solution: a dynamic programming algorithm

Defense: Correctness proof, complexity analysis

Systems

Problem: context-switches for kernel calls

Solution: move certain calls into user space

Defense: experiment to count kernel calls, context switches

Methodology

Problem: designing a data warehouse

Solution: a methodology for warehouse schema design

Defense: user study, compelling examples, "yardstick"

Variation on the Pattern

The "yardstick"

1. Problem
2. List of requirements for good solution
3. Where other solutions fail to meet the requirements
4. Your solution
5. Discussion of how your solution meets the requirements.

The most common failing in student papers is failing to present the problem

The next most-common failing is failing to explain *why* the problem is a problem

Survey Papers

Survey paper probably doesn't follow problem–solution–defense form.

Other forms:

Chronological

problem (solution, limitation)$^+$

e.g., string matching

brute force

failure function

reverse matching

Lines of development—several parallel sections

- Join optimization
  1. Dynamic programming
     - vines
     - bushy trees
  2. Transformational methods
     - memoization
     - search pruning
  3. Randomized methods
     - simulated annealing
     - genetic algorithms

Problem subclasses

(subclass characterization, solution)$^{+}$

Shortest path

- 1 source, 1 sink
- 1 source to all nodes
- all pairs

- positive edge weights
- 0-1 edge weights
- unrestricted edge weights

- sparse graphs

- parallel solutions

Experimental Papers—Special Considerations

Must give sufficient description of your experimental procedure

- for replication, and
- so someone can see why they got different results

  Environment

  Software used

  Test data

  Test procedure

Some of this material might go in an appendix or a "digital supplement"

Performance Section

If you report on a performance study, state its purpose, *e.g.*:

- Sensitivity analysis
- Model validation
- Comparison of methods
- Algorithm tuning
- Determining range of utility

Tell the reader what questions you seek to answer

Results

Findings, and brief discussion comparing your findings to previous work.

- Organize during writing by listing key items, gathering key graphics, gathering key equations.
- The order of presentation of the results should be the same as that of the technical questions in the introduction

  (Parallel structure)

Results

Core paragraph can include recap of your investigative purpose, reference to principal graphic that summarizes findings.

Arrange paragraphs by descending significance. Paragraphs progress from general to more specific.

Put key findings in lead position of paragraphs. Sentences progress from general to more specific.

Explain and interpret key results from figures, tables, equations and theorems.

Don't just say: the result are shown in Table 3.
Tell the reader *what to find* in Table 3.

Methods and Results

Sometimes, methods and results are intermingled. Particularly if several technical issues are tackled.

Method 1, Results from Method 1.

May raise new questions!

Method 2, Results from Method 2.

Method 3, Results from Method 3.

Discussion and Conclusion

Sometimes discussion and conclusion are separate sections, sometimes merged (particularly in conference papers).

- Can recap primary methods and results.
- Present major conclusions.
- Explains how results relate to prior work (previously mentioned in literature review).
- Generally *doesn't* introduce new literature or figures.
- Paragraphs are ordered according to descending significance — except for …

Discussion and Conclusion

Summary paragraphs at end.

Restates major conclusions and implications of results.

Good place to point out limitations of your work, and directions for extension: future work.

Discussion

Don't waste the discussion section!

Good discussion sections are hard to write.

Take a step back from your paper. Discuss with colleagues.

- Useful lessons
- Consequences
- Applicability elsewhere
- Follow-on work in progress, or perhaps published.
- Recommendations for similar endeavors (If only I knew then ...)

What do you most want the reader to remember?

- Speculation is OK in the discussion.
  - But say so — "Presumably ...", "We expect ..."

Acknowledgements

Be generous! Acknowledge:

- Colleagues you've discussed work with.
- Non-authors who provided materials (data, graphics, simulators, code).
- Referees who made comments and helped you improve your manuscript.
- Manager (in an industrial setting).
- Grant agencies (mention grants by number).
- Initiator of idea.

Acknowledgment is free. You don't know when you may want those people to help you again

Backmatter

Reference List

Alphabetical, or in order of mention, depending on required style.

Don't list references not cited in paper (*not* a bibliography).

Appendices

Supporting material that does not fit comfortably in the main text.

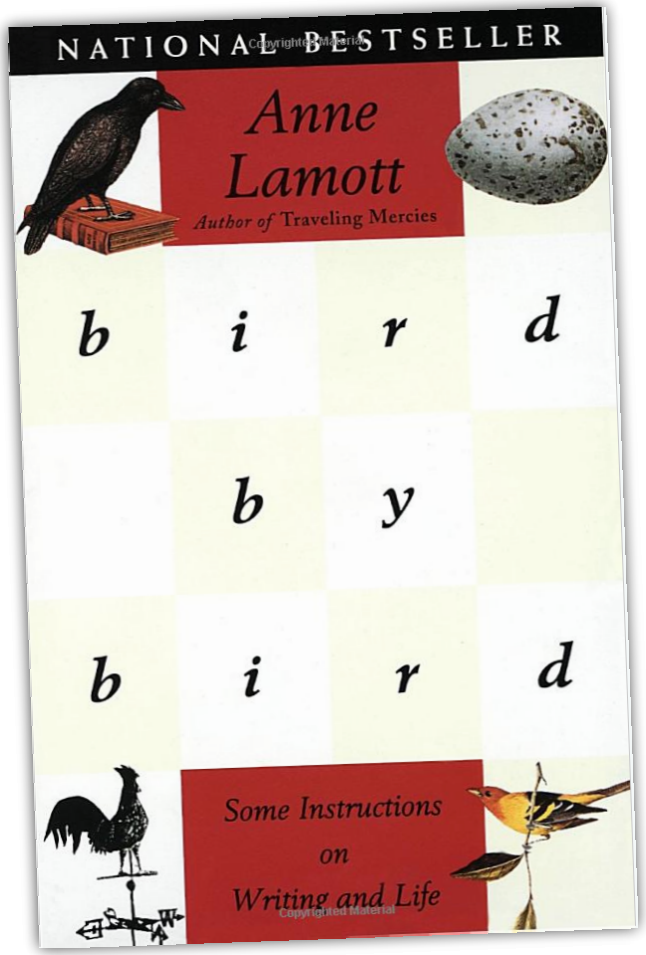
> Could include lengthy proofs, syntax & grammar of a language, code example, supporting data.

Call out appendices from the text (like figures).

Usually referred to by letters rather than numbers

We provide a complete syntax for the language in Appendix A

A lesson from fiction



Bird by Bird:
Some Instructions on Writing and Life

Anne Lamott

Lastly: I heard Alice Adams give a lecture on the short story once, one aspect of which made the writing students in her audience so excited that I have passed it along to my students ever since. (Most of the time I give her credit.) She said that sometimes she uses a formula when writing a short story, which goes ABDCE, for Action, Background, Development, Climax, and Ending. You begin with action that is compelling enough to draw us in, make us want to know more. Background is where you let us see and know who these people are, how they've come to be together, what was going on before the opening of the story. Then you develop these people, so that we learn what they care most about. The plot—the drama, the actions, the tension—will grow out of that. You move them along until everything comes together in the climax, after which things are different for the main characters, different in some real way. And