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
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
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!
The following guidelines on writing an abstract were given by Kent Beck at a Panel session at OOPSLA ‘93
(http://www.acm.org/sigplan/oopsla/oopsla96/how93.html)

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.
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.
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.
- Try to 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).

Andrew 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?
Introduction

Typical structure of an introduction:

1. Problem Statement
   • Situation
   • Problem sentence(s)
   • 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 are 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 important results and how they contrast with previous work.
Introduction

The entire *Problem Statement* tells the reader

1. What the problem is that you solve
2. Why the problem is important
3. Where the problem arises
4. What are the benefits and characteristics of a good solution (efficiency, accuracy, robustness, theoretical understanding)
5. The current status of the problem
6. The form previous solutions take (literature review)
7. The approach you’re taking, and how it’s motivated
8. What the new approach accomplishes
Introduction

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.)

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.

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 ...
Introduction

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
Introduction

Literature Review

Be careful about statements like

However, 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.
Introduction

Literature Review — Related Work

What constitutes relatedness?

- similar problem
- similar approach
- alternative investigation

Important to 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 is that a well-versed reader can reproduce your results. (More challenging for experimental than 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:

Theoretical

Problem: need for increased concurrency
Solution: new locking protocol

Defense:
Other Examples

Algorithms
Problem: 2-D pattern matching
Solution: a dynamic programming algorithm
Defense:

Systems
Problem: context-switches for kernel calls
Solution: move certain calls into user space
Defense:

Methodology
Problem: designing a data warehouse
Solution: a methodology for warehouse schema design
Defense:
Variation on the Pattern

• Problem
• List of requirements for good solution
• Where other solutions fail to meet the requirements
• Your solution
• Discussion of how your solution meets the requirements.
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
One More Form for Survey Papers

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
Special Considerations for Experimental Papers

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
Performance Section

If you report on a performance study, state its purpose

- Sensitivity analysis
- Model validation
- Comparison of methods
- Algorithm tuning
- Determining range of utility
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 reflect the order of the technical questions in the introduction.
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.
Sometimes methods and results are intermingled. Particularly if several technical issues are tackled.
Discussion and Conclusion

Sometimes separate sections, sometimes merged (particularly in conference papers).

Can recap primary methods and results.

Major conclusions presented.

Explains how results relate to review of literature.

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.
Don’t waste this section.
Good discussion sections are hard to write.
Take a step back from your paper for overview
  Useful lessons
  Consequences
  Applicability elsewhere
  Follow-on work in progress, or perhaps published.
  Recommendations for similar endeavors (If only I knew now ...)
What do you most want the reader to remember?
Speculation is OK here.
  (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.
Reference List

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

Don’t list references not cited in paper.

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 (“Appendix A”)