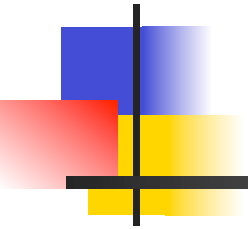


How to write a great research paper

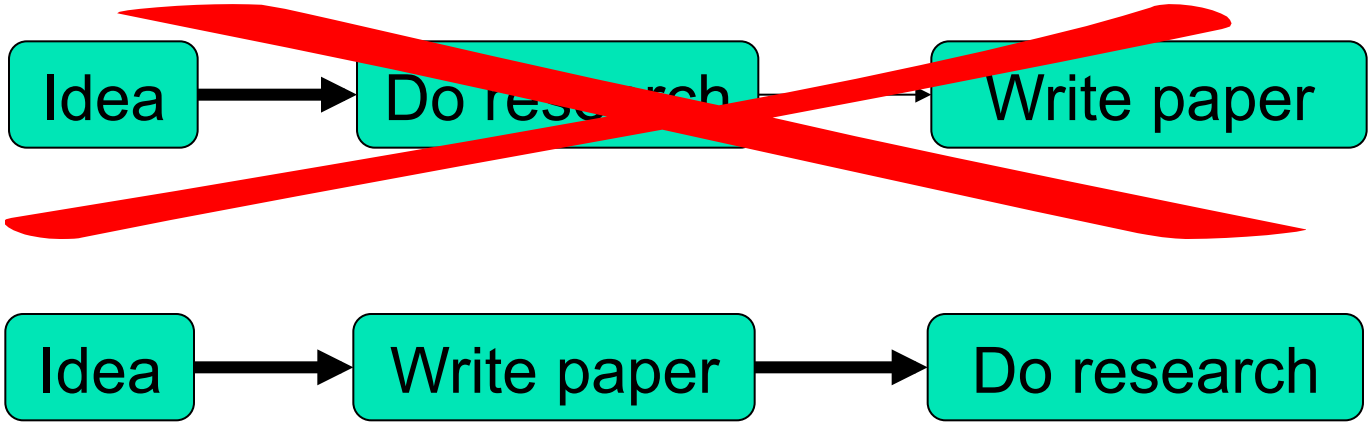
Adapted from Simon Peyton Jones
Microsoft Research, Cambridge

Writing papers is a skill

- Many papers are badly written
- Good writing is a skill you can learn
- It's a skill that is worth learning:
  - You will get more brownie points (more papers accepted etc)
  - Your ideas will have more impact
  - You will have better ideas

Writing papers: model 1

Writing papers: model 2



- Forces us to be clear, focused
- Crystallises what we don't understand
- Opens the way to dialogue with others: reality check, critique, and collaboration

Do not be intimidated

Fallacy You need to have a fantastic idea before you can write a paper. (Everyone else seems to.)

Write a paper,
and give a talk, about

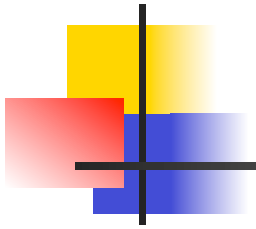
any idea,

no matter how weedy and insignificant it may seem to you

Do not be intimidated

Write a paper, and give a talk, about any idea, no matter how insignificant it may seem to you

- **Writing the paper is how you develop the idea in the first place**
- It usually turns out to be more interesting and challenging that it seemed at first

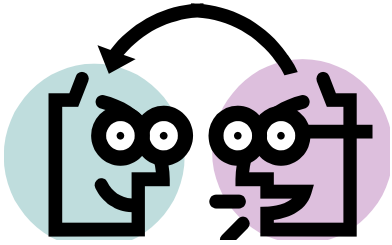


The purpose of your paper

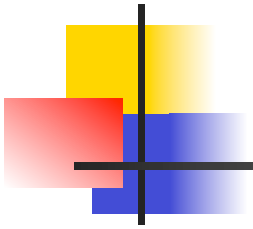
- Asking questions such as:
  - What problem did you want to solve?
  - Why is the problem important?
  - Which results are most interesting?
  - What is it that others can adopt?

Papers communicate ideas

- Your goal: to infect the mind of your reader with **your idea**, like a virus
- Papers are far more durable than programs (think Mozart)



The greatest ideas are (literally) worthless if you keep them to yourself



The Idea

Idea
A re-usable insight, useful to the reader

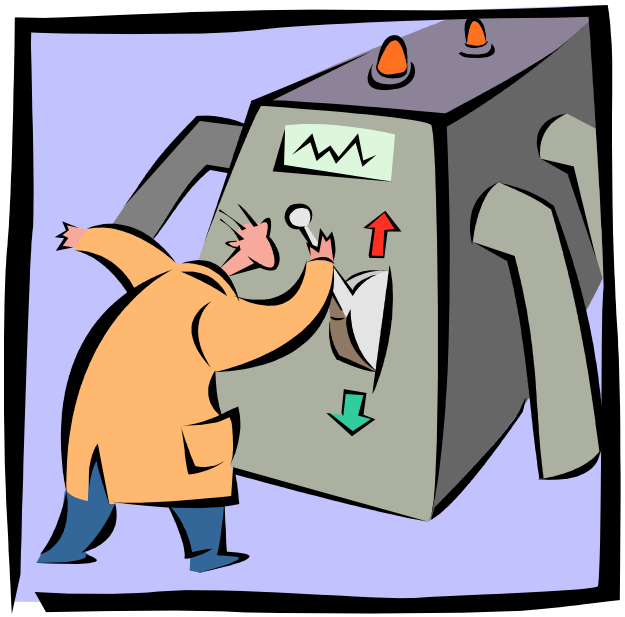
- Figure out what your main idea is
- Make certain that the reader is in no doubt what the idea is. Be 100% explicit:
  - "The main idea of this paper is...."
  - "In this section we present the main contributions of the paper."
- Many papers contain good ideas, but do not distil what they are.

One ping

- Your paper should have just one “ping”: one clear, sharp idea
- Read your paper again: can you hear the “ping”?
- You may not know exactly what the ping is when you start writing; but you must know when you finish
- If you have lots of ideas, write lots of papers

The purpose of your paper is not…

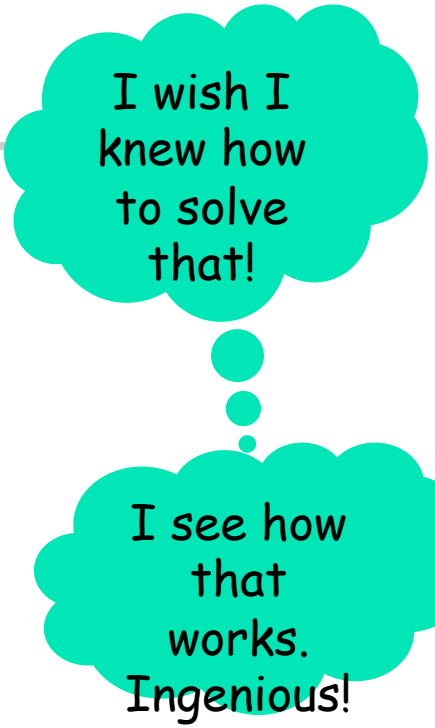
To describe the WizWoz system



- Your reader does not have a WizWoz
- She is primarily interested in re-usable brain-stuff, not executable artefacts

Your narrative flow

- Here is a problem
- It's an interesting problem
- It's an unsolved problem
- **Here is my idea**
- My idea works (details, data)
- Here's how my idea compares to other people's approaches



Structure (conference paper)

- Title (5-10 words, 1000 readers)
- Abstract (4 sentences, 100 readers)
- Introduction (1 page, 100 readers)
- The problem (1 page, 10 readers)
- My idea (2 pages, 10 readers)
- The details (5 pages, 3 readers)
- Related work (1-2 pages, 10 readers)
- Conclusions and further work (0.5 pages, 10 readers)

Title and Author

- Choose an accurate and simple title
  - A New Signature File Scheme Based on Multiple-Block File Descriptors Files for indexing Very Large Databases
  - Signature File Indexes based on Multiple-Block File Descriptors
- The convention in CS is not to give position, title or qualification
- Use the same style for names in all your papers so that it can be indexed consistently (Anne Ngu verses Anne H. Ngu..)
- Include a date to keep track of when the paper is completed.

The abstract

- I usually write the abstract last
- Used by program committee members to decide which papers to review
- Is a summary of what the paper is about.
- Four element style:
  1. State the problem
  2. Say why it's an interesting/challenging problem
  3. Say what your solution achieves
  4. Say how your solution was evaluated and what the outcome of the evaluation was

The introduction (1 page)

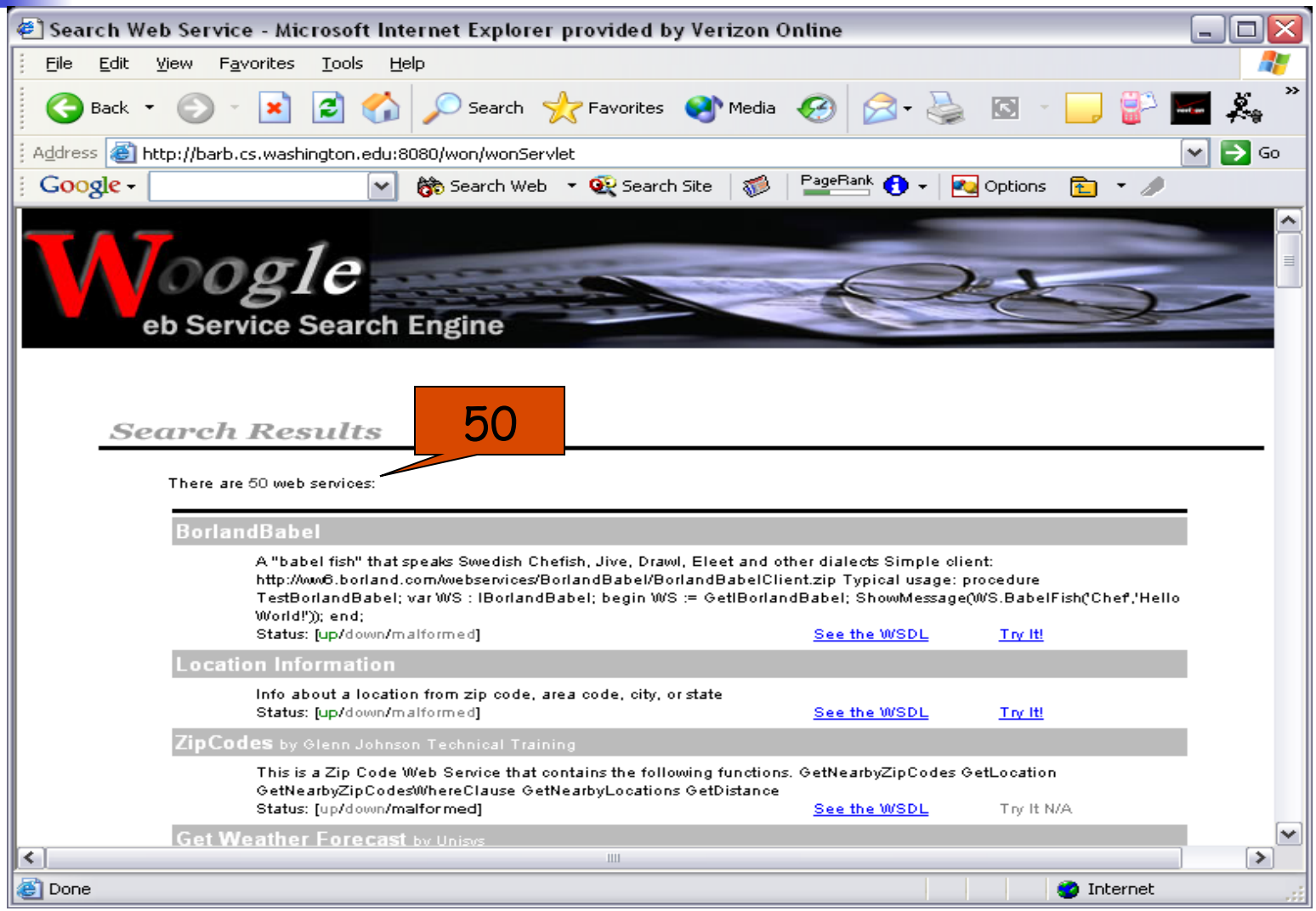
1. Describe the problem (problem motivation)
2. The approach to the problem
3. The scope and limitations of your solution
4. State your contributions (main outcomes)
5. Organization of the paper
6. It should omit detail, jargon and mathematics

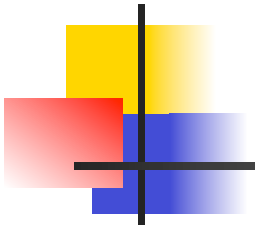
Describe the problem

We have also designed the brokering service so that it could be distributed among multiple brokers. Brokering could be provided by a single process or agent that saves all agent advertisements in a single repository, and processes all requests for services. In our experience, this architecture works well for agent systems with a few dozens agents. However, the broker then represents a single point of failure and a limit to scalability.

In InfoSleuth, we implemented a **distributed multibrokering** approach where many brokers collaborate to provide brokering services. In the distributed multibrokered system, many broker agents collectively maintain the information about agents in the system. Each agent in the system must be known to at least one broker, and a given agent may advertise its capabilities to one or more brokers,depending on the level of reliability it requires. When a broker receives a query for services, it not only searches for matches within its own repository, but also propagates the query to other relevant brokers, and collects the results together before returning them to the agent that sent the query. This approach is both robust and scalable.

Keyword Search does not Capture the Underlying Semantics



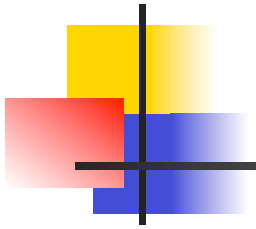


State your contributions

- Write the list of contributions first
- The list of contributions drives the entire paper: the paper substantiates the claims you have made
- Reader thinks "gosh, if they can really deliver this, that's be exciting; I'd better read on"

Introduction (Cont.)

- By the end of the introduction, the reader should understand the scope of your work.
- For example, if the topic is "Mechanism for Collaborative Authoring", then:
  - Who is doing the authoring?
  - What constraints are they working under?
  - What kinds of tasks are they trying to complete?
  - How sophisticated do the mechanism need to be?

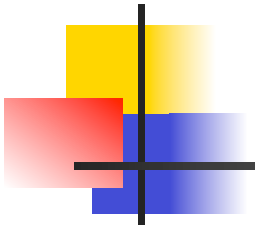


Contributions should be refutable

NO!	YES!
We describe the WizWoz system. It is really cool.	We give the syntax and semantics of a language that supports concurrent processes (Section 3). Its innovative features are...
We study its properties	We prove that the type system is sound, and that type checking is decidable (Section 4)
We have used WizWoz in practice	We have built a GUI toolkit in WizWoz, and used it to implement a text editor (Section 5). The result is half the length of the Java version.

Structure

- Abstract (4 sentences)
- Introduction (1 page)
- ~~Related work~~
- The problem (1 page)
- My idea (2 pages)
- The details (5 pages)
- Related work (1-2 pages)
- Conclusions and further work (0.5 pages)

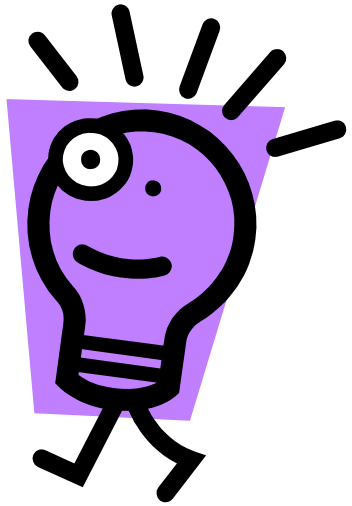


Background

- The background introduces the reader to:
  - Why the problem is interesting?
  - The technologies or concepts the paper's contribution is based on
  - Benchmark or baselines the contribution should be compared to
  - What other people have said about the same problem

No related work yet!

Related work



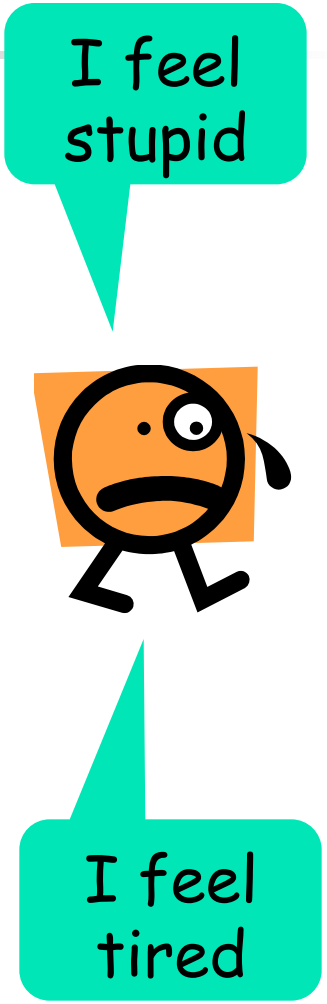
Your reader

Your idea

We adopt the notion of transaction from Brown [1], as modified for distributed systems by White [2], using the four-phase interpolation algorithm of Green [3]. Our work differs from White in our advanced revocation protocol, which deals with the case of priority inversion as described by Yellow [4].

No related work yet

- Problem 1: the reader knows nothing about the problem yet; so your (carefully trimmed) description of various technical tradeoffs is absolutely incomprehensible
- Problem 2: describing alternative approaches gets between the reader and your idea



Structure

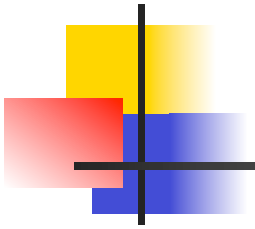
- Abstract (4 sentences) about 200 words
- Introduction (1 page) 1.5 pages
- The problem (1 page) 0.5 pages
- My idea (2 pages) overview of approach 1 page
- The details (5 pages) technical 2 pages, experiment (2-3 pages)
- Related work (1-2 pages) 1.5 pages
- Conclusions and further work (0.5 pages)

Presenting the idea in the body of a paper

3. The idea

Consider a bifircuated semi-lattice D, over a hyper-modulated signature S. Suppose p_i is an element of D. Then we know for every such p_i there is an epi-modulus j, such that $p_j < p_i$.

- Sounds impressive...but
- Sends readers to sleep
- In a paper you MUST provide the details, but FIRST convey the idea



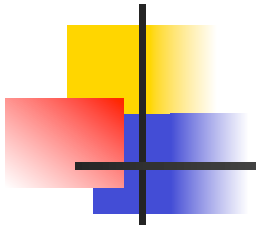
Presenting the idea

- Explain it as if you were speaking to someone using a whiteboard
- **Conveying the intuition is primary**, not secondary
- Once your reader has the intuition, she can follow the details (but not vice versa)
- Even if she skips the details, she still takes away something valuable

Putting the reader first

- **Do not** narrate your personal journey of discovery. This route may be soaked with your blood, but that is not interesting to the reader.

- Instead, choose the most direct route to the idea.



The payload of your paper

Introduce the problem, and your idea, using

EXAMPLES

and only then present the general case

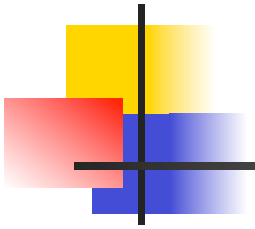
1) Provide Similar WS Operations

- Op1: GetTemperature
  - Input: Zip, Authorization
  - Output: Return
- Op2: WeatherFetcher
  - Input: PostCode
- Output: TemperatureF, WindChill, Humidity

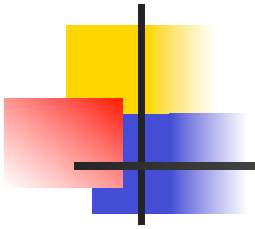
Similar Operations

→ Select the most appropriate one

Structure of the body of the paper (an example)

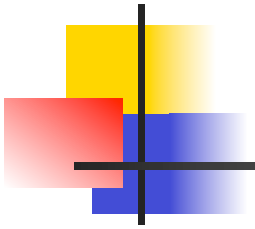


- External sorting
- Compression techniques for databse systems
- Sorting with compression
- Experimental setup
- Results and discussion



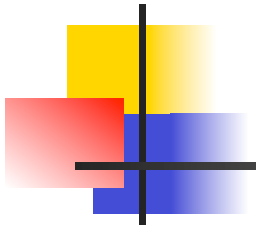
The details: evidence

- Your introduction makes claims
- The body of the paper provides **evidence to support each claim**
- Check each claim in the introduction, identify the evidence, and forward-reference it from the claim (state why it is novel)
- Evidence can be: analysis and comparison, theorems, experimental measurements, case studies



Structure

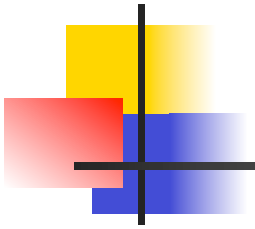
- Abstract (4 sentences)
- Introduction (1 page)
- The problem (1 page)
- My idea (2 pages)
- The details (5 pages)
- **Related work** (1-2 pages)
- Conclusions and further work (0.5 pages)



Related work

Fallacy To make my work look good, I have to make other people's work look bad

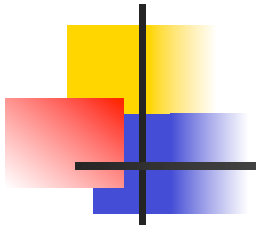
Need to explain a few sentences about each related work and how yours differ from them.



The truth: credit is not like money

Giving credit to others does not diminish the credit you get from your paper

- Warmly acknowledge people who have helped you
- Be generous to the competition. "In his inspiring paper [Foo98] Foogle shows.... We develop his foundation in the following ways..."
- Acknowledge weaknesses in your approach

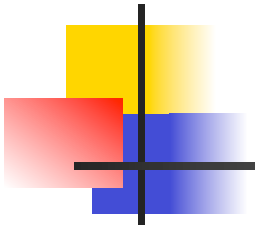


Credit is not like money

Failing to give credit to others can kill your paper

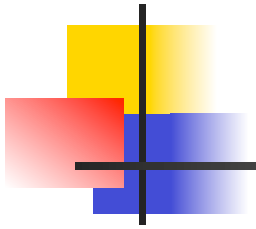
If you imply that an idea is yours, and the referee knows it is not, then either

- You don't know that it's an old idea (bad)
- You do know, but are pretending it's yours (very bad)



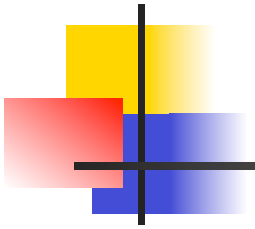
Structure

- Abstract (4 sentences)
- Introduction (1 page)
- The problem (1 page)
- My idea (2 pages)
- The details (5 pages)
- Related work (1-2 pages)
- Conclusions and further work (0.5 pages)



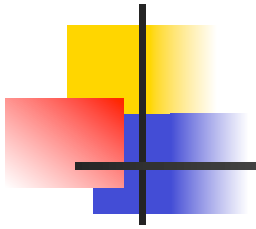
Conclusions and further work

- Be brief.
- Re-state your main idea
- Remaining challenges
- Future plan



Bibliography

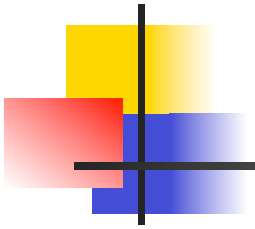
- A list of references that includes, papers, reports, books, theses, URLs cited in the text.



Appendices

- Used for bulky materials like listing of computer programs, details of proofs or experimental results.

The process of writing



Authorships

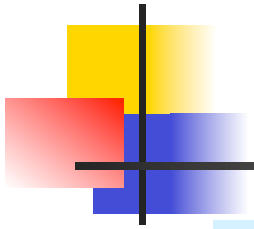
- Agree on a process for co-authoring:
  - Does everyone write a separate section? Who will put them all together?
  - Alternatively, each person contributes to the whole document, with each person taking a turn
  - Is one author allowed to delete another authors's text?

The process

- Start early. Very early.
  - Hastily-written papers get rejected.
  - Papers are like wine: they need time to mature
- Collaborate
- Use CVS/dropbox/GoogleDoc to support collaboration
- Break down the paper in different parts (easy to do that in LaTex)

First Draft

- Write freely for the first draft.
- Do not worry about the style
- Write down in point form what have been learnt, what have been achieved and what the results are.
- Good to have a skeleton structure defined
- First draft can be sloopy and you revise and edit later



Getting help

Get your paper read by as many friendly guinea pigs as possible

- Experts are good
- Non-experts are also very good
- Each reader can only read your paper for the first time once! So use them carefully
- Explain carefully what you want ("I got lost here" is much more important than "Jarva is mis-spelt".)

Getting expert help

- A good plan: when you think you are done, send the draft to the competitor saying "could you help me ensure that I describe your work fairly?".
- Often they will respond with helpful critique (they are interested in the area)
- They are likely to be your referees anyway, so getting their comments or criticism up front is Jolly Good.

Listening to your reviewers

Treat every review like gold dust

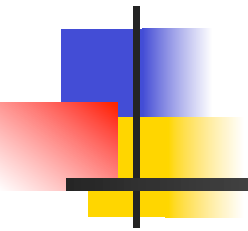
Be (truly) grateful for criticism as well as praise

This is really, really, really hard

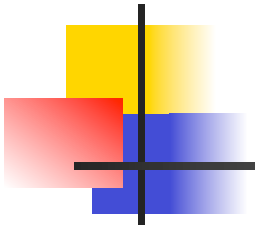
But it's
really, really, really, really, really, really, really, really, really, really
important

Listening to your reviewers

- Read every criticism as a positive suggestion for something you could explain more clearly
- DO NOT respond "you stupid person, I meant X". Fix the paper so that X is apparent even to the stupidest reader.
- Thank them warmly. They have given up their time for you.

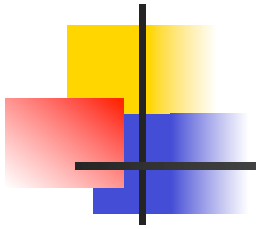


Language and style



Basic stuff

- Submit by the deadline
- Keep to the length restrictions
  - Do not narrow the margins
  - Do not use 6pt font
  - On occasion, supply supporting evidence (e.g. experimental data, or a written-out proof) in an appendix
- Always use a spell checker



Visual structure

- Give strong visual structure to your paper using
  - sections and sub-sections
  - bullets
  - italics
  - laid-out code
- Find out how to draw pictures, and use them

Visual structure

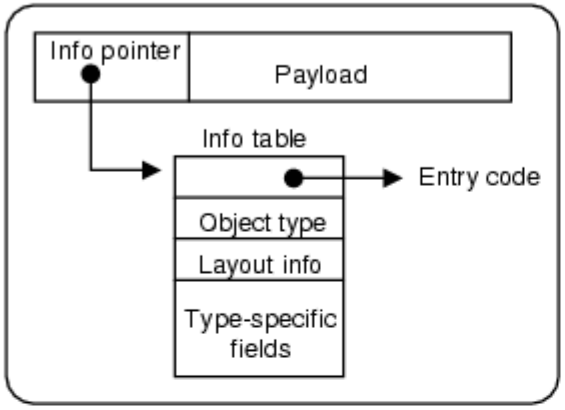


Figure 3. A heap object

The three cases above do not exhaust the possible forms of f. It might also be a *THUNK*, but we have already dealt with that case (rule THUNK). It might be a *CON*, in which case there cannot be any pending arguments on the stack, and rules UPDATE or RET apply.

4.3 The eval/apply model

The last block of Figure 2 shows how the eval/apply model deals with function application. The first three rules all deal with the case of a *FUN* applied to some arguments:

- If there are exactly the right number of arguments, we behave exactly like rule KNOWNCALL, by tail-calling the function. Rule EXACT is still necessary — and indeed has a direct counterpart in the implementation — because the function might not be statically known.
- If there are too many arguments, rule CALLK pushes a *call*

remainder of the object is called the *payload*, and may consist of a mixture of pointers and non-pointers. For example, the object $CON(C\ a_1 \ldots a_n)$ would be represented by an object whose info pointer represented the constructor C and whose payload is the arguments $a_1 \ldots a_n$.

The info table contains:

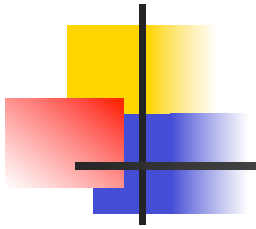
- Executable code for the object. For example, a *FUN* object has code for the function body.
- An object-type field, which distinguishes the various kinds of objects (*FUN*, *PAP*, *CON* etc) from each other.
- Layout information for garbage collection purposes, which describes the size and layout of the payload. By "layout" we mean which fields contain pointers and which contain non-pointers, information that is essential for accurate garbage collection.
- Type-specific information, which varies depending on the object type. For example, a *FUN* object contains its arity; a *CON* object contains its constructor tag, a small integer that distinguishes the different constructors of a data type; and so on.

In the case of a PAP, the size of the object is not fixed by its info table; instead, its size is stored in the object itself. The layout of its fields (e.g. which are pointers) is described by the (initial segment of) an argument-descriptor field in the info table of the FUN object which is always the first field of a PAP. The other kinds of heap object all have a size that is statically fixed by their info table.

A very common operation is to jump to the entry code for the object, so GHC uses a slightly-optimised version of the representation in Figure 3. GHC places the info table at the addresses *immediately*

Use simple, direct language

NO	YES
The object under study was displaced horizontally	The ball moved sideways
On an annual basis	Yearly
Endeavour to ascertain	Find out
It could be considered that the speed of storage reclamation left something to be desired	The garbage collector was really slow



Summary

If you remember nothing else:

- **Identify your key idea**
- **Make your contributions explicit**
- **Use examples**

A good starting point:

"Advice on Research and Writing"

http://www-2.cs.cmu.edu/afs/cs.cmu.edu/user/mleone/web/how-to.html

Use the active voice

The passive voice is "respectable" but it DEADENS your paper. Avoid it at all costs.

NO	YES
It can be seen that...	We can see that...
34 tests were run	We ran 34 tests
These properties were thought desirable	We wanted to retain these properties
It might be thought that this would be a type error	You might think this would be a type error

"We" = you and the reader

"We" = the authors

"You" = the reader