# CPSC 365 Proofs Class 1

James Williams

January 19, 2015

## 1 Background

### 1.1 Types

Suppose that we wish to prove the premise $A$ implies the result $B$, or $A \rightarrow B$. In the context of problem solving, the premise $A$ is all of the information that can be extracted from the problem, lecture notes, and chapters, and the result $B$ is the statement that needs to be proved.

#### Trivial proof

When something doesn't need to be proved, it's trivial. Formally, the result $B$ is true regardless of the premise $A$, so $A \rightarrow B$ is trivial. Note that stating such a result can still be important, rather than leaving it for the reader to determine, even if it doesn't need to be proved.

#### Direct proof

When there is a direct line of reasoning from the premise $A$ to the result $B$, or $A \rightarrow B$. This type of proof is hard to write well, as getting the line of reasoning correct, clear, and concise can be tricky. Also, constructing the steps of the proof directly can often introduce more steps than needed.

Also, even when the direct line of reasoning from the premise $A$ to the result $B$ is so obvious that it seems trivial, it isn't trivial. Something still needs to be proved, and you still need to state the reasoning!

#### Indirect proof (contradiction / contrapositive)

When a direct line of reasoning is difficult to prove, sometimes it can be easier to assume the premise $A$ and the negation of the result $\neg B$ and to derive a contradiction between $A$ and $\neg B$. From this, it follows that $A \rightarrow B$ as desired. A special case of this method is the contrapositive, where you assume the negation of the premise $\neg A$ and prove the negation of the result $\neg B$, or $\neg A \rightarrow \neg B$. From this, the contrapositive $A \rightarrow B$ follows as well.

Note that you should not use contradication as a crutch, stumbling blindly around the negation of the result and trying to coerce the contradiction required. This will get you into trouble, as it is easy to make mistakes that will lead to a contradiction that is invalid.

## 1.2 Methods

**Proof by induction (very handy)**

Sometimes, you want to prove more than just one thing. To prove the logical statement $S(n)$ for any $n \in \mathbb{N}$, we prove that the statement $S(1)$ is true for $n = 1$, and that the premise $S(k)$ implies the result $S(k+1)$ for any $k \in \mathbb{N}$, or $S(k) \to S(k+1)$ for any $k \in \mathbb{N}$. From this, it follows that the statement $S(n)$ is true for any $n \in \mathbb{N}$.

For example, suppose that someone asks you to push over $n$ dominoes for any $n \in \mathbb{N}$. Now, you could set them up side by side and push them over one by one, until you get to $n$ or get bored. However, you could set them up end to end, check that the first domino can be pushed, check that the distance between dominoes is less than the height of dominoes, push the first domino, and sit back and enjoy the show.

**Example**

Prove that the partial sum formula for triangular numbers

$$S(n): \quad 1+2+3+\cdots+n = \frac{n(n+1)}{2}$$

holds for all $n \in \mathbb{N}$.

First, we prove that the formula $S(1)$ holds for $n = 1$. For $n = 1$, LHS $= 1$ and RHS $= \frac{1}{2}(1+1) = 1$, and LHS $=$ RHS as desired.

Next, we assume that the formula $S(k)$ holds for some $k \in \mathbb{N}$

$$S(k): \quad 1+2+3+\cdots+k = \frac{k(k+1)}{2}$$

and then use this to prove that the formula $S(k+1)$ holds for $k+1$ as well. Adding $(k+1)$ to both sides of $S(k)$

$$\begin{aligned} 1+2+3+\cdots+k+(k+1) &= \frac{k(k+1)}{2}+(k+1) \\ &= \frac{k(k+1)+2(k+1)}{2} \\ &= \frac{(k+1)(k+2)}{2} \\ &= \frac{(k+1)((k+1)+1)}{2} \end{aligned}$$

which gives $S(k+1)$

$$S(k+1): \quad 1+2+3+\cdots+(k+1) = \frac{(k+1)((k+1)+1)}{2}$$

as desired.

From this, it follows by induction that the formula $S(n)$ holds for all $n \in \mathbb{N}$.

## 1.3 Styles

After the logical argument has been decided and the problem solved, there are many ways that you can actually write up your proof, and not all of them are good. This will be expanded in Sections 1.4 and 3.1, but for now we outline three generic styles that you should have come across already.

***Two column***

- Think high school geometry, where you have one column of logical steps and one column of detailed reasoning.
- The proof of the illiterate, neither prose nor poetry.

***Crayon***

- When your proof covers many pages, and you have to think really hard to remember how the pages should be ordered.
- You might have the actual solution, but it requires a fairly detailed legend.
- Typically, your rough work will start out this way, and *you* need to translate it into something more readable!

***Paragraph***

- Concise and well constructed, your proof reads like prose, but has the elegance of poetry.
- You shouldn't write *everything* in paragraphs, but use diagrams and display formatting to improve clarity and simplicity.
- This is the starting point for a great proof!

***Your style***

- Use a blend of different types, styles, and method as appropriate, neither conforming exclusively to one school of thought nor changing between schools of thought arbitrarily.
- When you discover a particularly useful tool, use it consistently, so that the reader develops an understanding of the structure and method of your work in general.
- Develop your style, and make it personal. You should always be improving!

## 1.4 Tools

**Notation / terminology**

Using precise notation and correct terminology can make a proof much more straightforward, both to write and to read. Mathematical symbols, set notation, logic symbols, and shorthand help make arguments and descriptions less verbose and more precise, which is essential.

Follow convention as much as possible, as it exists for a reason! For example, use lower case letters for vectors and scalars, capital letters for matrices, greek symbols for angles, coefficients, or parameters, and avoid using the letters $l$, $o$, and $O$ as they are hard to distinguish from 1 and 0.

The following tables contain a selection of terminology, particularly common practice for naming variables and other entities in mathematics and graph theory. Note while that these conventions do not have to be followed, they are fairly universal, and make the working easier to decipher and understand.

| | |
|---|---|
| variables | $x$, $y$, $z$ |
| matrices | $A$, $B$, $C$, $D$, $H$, $I$, $L$, $M$, $P$, $Q$, $R$, $S$, $T$, $X$, $Y$, $Z$ |
| vectors / points | $u$, $v$, $w$ |
| scalars / points | $a$, $b$, $c$ |
| functions | $f$, $g$, $h$ |
| parameters | $r$, $s$, $t$ |
| angles | $\theta$, $\phi$, $\psi$ |
| probabilities | $p$, $q$ |
| counters | $i$, $j$, $k$ |
| sizes | $n$, $m$, $N$, $M$ |
| scalar coefficients | $\alpha$, $\beta$, $\gamma$ |
| scalar parameters | $\mu$, $\lambda$, $\omega$, $\kappa$, $\phi$, $\varphi$, $\psi$ |
| small changes | $\epsilon$, $\delta$ |

Table 1: General mathematics terminology

| | |
|---|---|
| graphs / trees | $G$, $T$ |
| node sets | $U$, $V$ |
| edge sets | $E$ |
| adjacency matrices | $A$ |
| weight matrices | $W$ |
| nodes | $a$, $b$, $c$, $d$, $u$, $v$, $w$ |
| edges | $e$, $f$, $g$ |
| weights | $w$ |
| degrees | $k$, $d$ |

Table 2: Graph theory specific terminology

| | |
|---|---|
| basic relations | $=$, $\neq$, $\equiv$, $\sim$, $\simeq$, $>$, $<$, $\geq$, $\leq$, $\gg$, $\ll$ |
| basic operations | $+$, $-$, $\pm$, $\mp$, $\times$, $\cdot$, $\div$, $/$, $\wedge$, $\vee$, |
| set relations | $\in$, $\notin$, $\supset$, $\subset$, $\not\supset$, $\not\subset$, $\supseteq$, $\subseteq$, $\not\supseteq$, $\not\subseteq$ |
| set operations | $\cap$, $\cup$, $\setminus$, $\times$ |
| membership | $\{\ \}$, $(\ )$ |
| logical | $\therefore$, $\rightarrow$, $\leftarrow$, $\leftrightarrow$, $\Rightarrow$, $\Leftarrow$, $\Leftrightarrow$, $\exists$, $\exists!$, $\nexists$, $\forall$ |

Table 3: Shorthand notation / terminology

Think of mathematics as a language, similar to programming, but somewhere in between prose and poetry. You should avoid describing everything using words and paragraphs, which cripples your proof with verbosity and redundancy, but at the same time, you should avoid excessive use of symbols and shorthand, which trades clarity and readability for compactness and brevity.

Also, you can find many useful resources and cheat sheets online, and a selection is provided in the bibliography.

**Diagrams**

Supplementing descriptions with diagrams can simply a proof significantly. Diagrams can be used to describe algorithms, processes, graph topology, or data structures, to sketch functions, densities, or distributions, and to define notation, symbols, and terminology.

Here are two simple examples from a problem set for Dan Spielman's *Graphs and Networks* CPSC 462 / 562

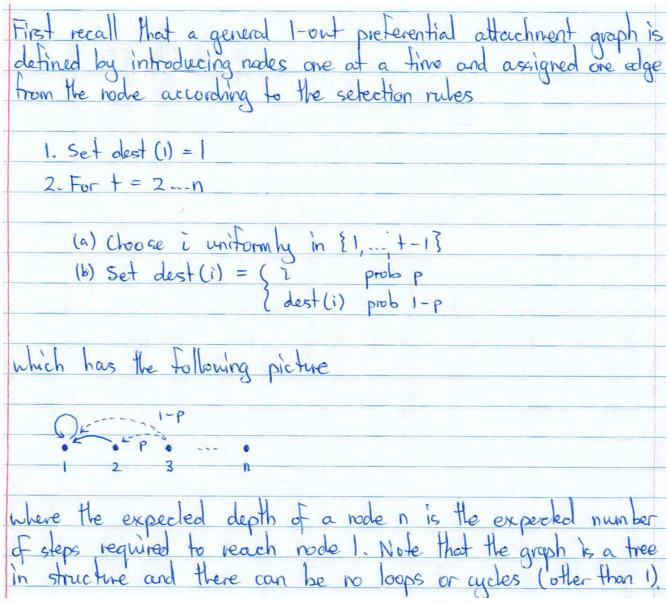


Figure 1: Drawing of a graph generating process (preferential attachment model)

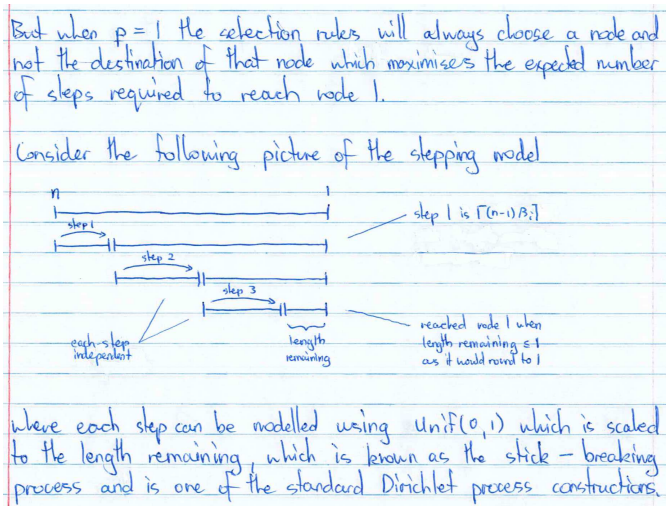


Figure 2: Drawing of the Dirichlet stepping process going from node $n$ to node 1

Note that it can be dangerous to use diagrams without the accompanying steps or working, or to gloss over errors in your working. *Proof by diagram* is only a suitable method for straightforward arguments, and is easily abused when the argument should be more formal.

### Components

Sometimes a problem naturally breaks down into distinct components, and the solution should be as well. Avoid breaking down a problem needlessly, as separating a straightforward logical argument into too many components will make it harder to read. However, when needed, there are several conventional components that can be used.

- Observation
  - typically an obvious or *self proving* statement for later reference, but stated formally
  - sometimes used as preliminary statements to structure the information provided
- Claim
  - similar to an observation in detail, but requires a straightforward proof as well
- Proposition
  - a supporting statement that is assumed, used, but not proved until later for coherence and clarity
  - can be used to make proving something else more straightforward
  - typically an intuitive or reasonable statement, but with possibly tedious formal detail
- Lemma
  - statement or result that is used only as a *stepping stone* to prove another result (typically a theorem)
- Theorem
  - the overall statement to be proved
  - think of a theorem as a problem and the proof as the solution (multiple problems, multiple theorems)
- Corollary
  - statement that follows from another statement with little or no further reasoning
  - *prove a theorem, get a corollary*
  - a straightforward generalisation of a simpler statement would be a corollary

There are different conventions for using these components, and while none of them are right or wrong, they all use these components specifically and consistently. In CPSC 365, the problem sets frequently break down some theory into distinct problems, none of which are intricate enough to require breaking down into more components.

## 2 Method

You should always have a plan of action when solving a problem or proving a theorem! I have elaborated on my general approach to solving problems, which breaks down into five steps and requires as much planning as solving.

### Steps

- UNDERSTAND
- PLAN
- SOLVE $\rightleftarrows$ FIX
- PLAN
- WRITE

**UNDERSTAND**

- rewrite the problem / write on the printed problem, elaborating, rephrasing, connecting, explaining
- identify the key components of the problem / the parts to be solved and then reassembled
- there is often a part that requires a spark of inspiration, and once this is solved, the problem is solved
- exploring some basic examples / counterexamples that illustrate the result can be helpful

**PLAN**

- plan out solution steps corresponding to the problem components
- establish the order of work / the importance of the steps
- don't agonize over fine detail until you are confident that your overall method is correct / reasonable
- draw diagrams and introduce notation, symbols, and terminology to make work more straightforward

**SOLVE $\rightleftarrows$ FIX**

- solve each step of the **plan**, and adapt the **plan** when required
- effectively, take the intuition and the concept of the plan and make the rough working more formal
- work until the proof is complete or some fundamental error is discovered
- sometimes, you need to throw everything out and go back to the drawing board / **understand**

**PLAN**

- when the proof is complete, the solution is coherent, and all steps are consistent, plan out the writing
- take the content and structure the proof / argument into an readable, organised, and logical narrative
- *think backwards, write forwards*
- this is probably the most important part of writing a good proof, once you have the problem solved

**WRITE**

- write the solution as to another student in the class with the same tools, knowledge, and background
- don't assume that the reader has the ability to solve or even understand the problem, explain it well
- aim equally for coherence and conciseness
- write prose with the elegance of poetry, and use diagrams, symbols, and terminology as appropriate

When working through these five steps, frequently going back to the drawing board, the solution evolves from small scribbles on the problem sheet through rough working spread over many pages to a tidy write up that spans only several pages. Trying to go directly from **understand** to **write** will end up with someone getting frustrated and probably annoyed, either you or the reader (who is the grader in CPSC 365). Note that there is a difference between a good proof and a bad proof, even when they are both a correct proof!

# 3 Summary

## 3.1 Comments

Here are some additional comments to keep in mind in general.

- writing layout and formatting (see also the Art of Problem Solving)
  - use margins, and stick to them!
  - whitespace is your friend
  - write horizontally and neatly
  - center / indent display math and diagrams
  - use convention for symbols / terminology
- general comments
  - plan really well
  - develop a concept of *mathematical elegance* and put it into practice
  - write prose with the elegance of poetry
  - learn from example / feedback (especially Dan Spielman's solutions)
  - when you prove something, try to find any flaws in your reasoning

## 3.2 References / Bibliography

- general references

| | |
|---|---|
| mathematical proof summary | goo.gl/EKq3bh |
| Art of Problem Solving *"How to Write a Solution"* | goo.gl/j8nL5y |
| methods of proof | goo.gl/OV22aC |
| general principles of mathematical communication | goo.gl/PS0Yur |

- symbols / terminology

| | |
|---|---|
| list of mathematical symbols | goo.gl/GJhIag |
| set notation | goo.gl/98Fku6 |
| list of logic symbols | goo.gl/1e5n4k |
| common mathematical symbols and abbreviations | goo.gl/EnU8YD |
| mathematical symbols | goo.gl/oDkbEz |

- symbols for LaTeX

| | |
|---|---|
| LaTeX math symbols | goo.gl/I6vnVf |
| Art of Problem Solving list of LaTeX symbols | goo.gl/rcXYVi |
| LaTeX mathematical symbols | goo.gl/bbEBrn |
| The Great, Big List of LaTeX Symbols | goo.gl/37ipBj |
| The Comprehensive List of LaTeX Symbols | goo.gl/2xJbz6 |

### 3.3 Examples

There are three handwritten examples attached, relating to Problem 1 of Problem Set 0.

### 3.4 D-day

My method for sloving a problem in a week (or one of Dan Spielman's problem sets):

- D-day $-7$ : Read the problem, feel stupid.
- D-day $-6$ : Review recent lectures / chapters, understand the problem, and plan out an strategy.
- D-day $-5$ : Solve the problem, start the proof.
- D-day $-4$ : Realize that my solution is wrong, start over. I still got time!
- D-day $-3$ : Fix my solution, write a new proof.
- D-day $-2$ : Put the finishing touches on my write up, run simulations, and prove a tighter bound.
- D-day $-1$ : Go party.
- D-day : Submit!
- D-day $+1$ : Discover a new, two line solution, kick myself, and feel stupid.

You have to think about problems 24 / 7, inspiration can hit at any time, and you don't want to miss it!

# Problem Set 0

## Direct Proof (BAD)

(2), (3) => (1)

T has $n-1$ edges, and is connected. Consider some $u \in V$. As T is connected, there exists some $v \in V$ such that $(u,v) \in E$. Let $C = u$, where $C \subseteq V$. Then we can add $v$ to $C$ such that $C = C \cup \{v\}$, where $C \subseteq V$, $E$ restricted to $C$ or $E_c$ contains only one edge $(u,v) \in E_c$ such that $|E_c| = 1$, and no cycles are introduced as the edge added corresponds to the node added, and is a bridge (where an edge cannot be a bridge and part of a cycle at the same time). Now, recursively, there exists some $w \in V$ such that $w \notin C$ (unless $|V \setminus C| = 0$ and $C = V$) and for some $x \in C$ there exists $(x,w) \in E$. Hence, we add $w$ to $C$ such that $C = C \cup \{w\}$, where $C \subseteq V$, $E_c$ now contains only one more edge $(x,w)$ and $|E_c| \mapsto |E_c| + 1$, and no cycles were introduced. As $n = |V|$, we can only add $n-1$ nodes to $C$ until $C = V$, and so $|E_c| = n-1$. Hence $(C, E_c) = (V, E) = T$ and T has no cycles (as no cycles were introduced).

## Indirect Proof (GOOD)

(2), (3) => (1)

From (2) and (3), $T = (V, E)$ has $|E| = n-1$ edges, and is connected.

Suppose that T has a cycle as well. Then $\exists\, e \in E$ that is not a bridge, such that $G = (V, E \setminus e)$ remains connected. $|E \setminus e| = n-2$ is trivial. However, as $G$ is still connected, $|E \setminus e| \geq |V| - 1 = n-1$, which is a contradiction.

Thus, T has no cycles, and (1) is proved.

## Induction Proof (Method)

(1), (3) => (2)

From (1) and (3), T has no cycles and is connected.

First, $|E| \geq n-1$ as T is connected. We will use induction to prove $|E| \leq n-1$ as T has no cycles (and is connected). Let $S(n)$ be the logical statement that "a directed, acyclic graph T with $|V| = n$ has at most $|E| \leq n-1$."

$S(0)$ : Trivial.

$S(1)$ : $n = 1$, and $V = \{v\}$. Then $E = \{\}$ and $|E| = 0 = 1-1$.

$S(k)$ : Assume $S(k)$.

Consider a directed, acyclic graph $G = (V, E)$ with $|V| = k+1$. For some $v \in V$ let H be the induced subgraph on $V \setminus \{v\}$, which has $t \geq 1$ connected components $H_i = (V_i, E_i)$ for $i = 1 \ldots t$. Each component is connected and acyclic (inherited from G) and $|V_i| \leq k$ for $i = 1 \ldots t$, and $S(k)$ implies that $|E_i| \leq |V_i| - 1$. Noting that

$$\sum_{i=1}^{t} |V_i| = k = |V| - 1$$

it follows that

$$\sum_{i=1}^{t} |E_i| \leq k - t$$

and as at most t edges were removed with v as G is acyclic and connected $|E| \leq k - t + t = k$ as well.

Thus, $S(k)$ implies $S(k+1)$.

Hence, it follows by induction that $S(n)$ is true for all $n \in \mathbb{N}$, and (2) is proved.