

# Coding Notes

Jeffrey Young

Feb 23, 2017

## Purpose

The Purpose of encodings is to systematically and formally codify real world, in the wild, explanations so that observing larger patterns becomes possible. The end goal of this is:

1. Generate a database of encodings
2. Analyze database to find patterns of explanation
3. 1 and 2 become formative work for exploring DSL possibilities in XOP

## Scope

The scope of the database is restricted to explanations of common Computer Science algorithms *from* Universities only. Restricting the scope in this manner provides two benefits:

1. All explanatory objects have a stated, intrinsic goal to communicate the mechanics, application, and implementation of similar things
2. There are numerous examples of different approaches to explain *the same* thing, and numerous examples of *like* approaches to explain different things

## Data Collection

All Data is coded by human individuals with reference to this document. The location index represents the location of an encoding *relative* to the document. So a location of 1 means the literal first page of the document (typically a header slide or introduction).

## Some Terminology

In our terminology an explanation is called an "explanation artifact", our working model of the process of explaining is a non-linear sequence of "steps", where each "step" denotes some progress in the understanding of the explanation artifact on the part of the information receiver. More formally:

**Explanatory Artifact** The whole explanation, including all the step taken in the explanation, the language used in the explanation etc.

**Step** The steps that are taken, in an explanatory artifact, that guide the reader from non-understanding to understanding.

# Syntax and Grammer

The syntax of an encoding is given by  $\_ - \_ - \_ - \_ \subseteq \text{Location} \times \text{Level} \times \text{Role} \times \text{Notation}$  where

$l \in \mathbb{N}$

$g \in \text{Level} ::= \text{Problem} \mid \text{Algorithm} \mid \text{Implementation}$

$r \in \text{Role} ::= \text{Background} \mid \text{Definition} \mid \text{Constraint} \mid \text{Example} \mid \text{Application} \mid \text{Variant} \mid \text{Analogy} \mid \text{Idea} \mid \text{Proof} \mid \text{Performance} \mid \text{Properties}$

$n \in \text{Notation} ::= \text{English} \mid \text{Math} \mid \text{Diagram} \mid \text{List} \mid \text{Table} \mid \text{Sequence} \mid \text{Pseudocode} \mid \text{Code} \mid \text{Animation} \mid \text{Picture} \mid \text{Plot} \mid \text{Empty} \mid \text{n/n}$

## Semantics

### Location

a location  $l$ , specifies the location that the encoding is referring to, this could be a slide, a number line etc.

### Level

a Level  $g$ , Specifies the level of abstraction for a given step, a level can be one of:

**Problem**  $\triangleq$  the purpose of a given step is to elucidate the motivating problem of the algorithm, i.e the problem that the algorithm will solve

**Algorithm**  $\triangleq$  the purpose of a given step is to explain the algorithm at hand

**Implementation**  $\triangleq$  the purpose of a given step is to explain the implementation details of the algorithm, e.g. What data structures to use, What the form of the code that implements the algorithm should be

### Role

a Role  $r$ , specifies how the Level is trying to be reached, denotes the answer to question such as "What is the step trying to convey?":

In general the meaning of each role is:

**Background**  $\triangleq$  A given level is reached by a step that describes the history, creators, genealogy of the Algorithm

**Definition**  $\triangleq$  A given level is reached by a step that explicitly provides a formal definition.

**Constraint**  $\triangleq$  A given level is reached by a step that explicitly presents a limit or condition in which the level would cease to be valid, e.g. Dijkstra's only works on non-negative weighted graphs

**Example**  $\triangleq$  A given level is reached by a step that provides an Example

**Application**  $\triangleq$  A given level is reached by a step that explains what the algorithm is useful for

**Variant**  $\triangleq$  A given level is reached by a step that describes things that are similar but slightly different than the algorithm. For example, describing Prim's algorithm and it's similarities to Dijkstra's or describing the similarities between a dog and a wolf

**Analogy**  $\triangleq$  A given level is reached by a step that provides an Analogy to explain the algorithm at hand. For example a visual analogy for Dijkstra's could be: If you have a physical model of a graph, and you pick it up by one vertex, then the vertex with the shortest path to the "source" vertex will be the one farthest from the ground.

**Idea**  $\triangleq$  A given level is reached by a step that adds an abstract idea to the explanation as a way to progress.  
For example, the statement "Well we have this, *what if we did* this?"

**Proof**  $\triangleq$  A given level is reached by a formal proof

**Performance**  $\triangleq$  A given level is reached by a step that explicitly describes the computational complexity of the level

**Properties**  $\triangleq$  A given level is reached by a step that explicitly describes the properties of that level. For Example, AVL Trees are both *balanced* and *ordered*.

Consider the following matrix of Level Role combinations of Dijkstra's algorithm. Not all of the cells will be orthogonal to each other. In this case we would have:

**Problem:** How to traverse the shortest path in a non-negative weighted graph

**Algorithm:** Dijkstra's Algorithm

**Implementation:** You should use a Priority Queue that has efficient lookup, mutate operations.

- $\perp$  used to denote cells which may be nonsensical

Role	Problem	Algorithm	Implementation
Definition	Mathematical definition of Problem	Mathematical Definition of Algorithm	$\perp$
Example	Display of a non-negative weighted graph	Showing the algorithms execution on the map	Showing requisite data structures etc.
Application	Real World Example of the problem	$\perp$	Triage System in a Hospital
Background	History of the Problem	History, Author, etc.	History of Priority Queues
Variant	Perhaps a teleporter exists, now what is shortest path	Description of Bellman-Ford	Description of slightly different Priority Queues
Analogy	$\perp$	Exposition of Prim's algorithm	$\perp$
Performance	$\perp$	Complexity	Complexity of requisite data structs
Idea	$\perp$	$\perp$	$\perp$
Constraint	Depiction of the Constraints of the Problem	Depiction of domain where Algorithm lacks validity	Requirements of internal Data Structs
Proof	$\perp$	Explicit Proof of Algorithm correctness	Explicit Proof of some requisite part of the algorithm

## Notation

a Notation  $n$ , specifies the form of the role, and can be one of:

**English**  $\triangleq$  Human language to give explanations/statements.

**Diagram**  $\triangleq$  Diagram in the manner of data structures, such as graph, list.

**List**  $\triangleq$  List of similar items

**OrderedList**  $\triangleq$  Step by step items

**Math**  $\triangleq$  Formulas/math style symbols.

**Pseudocode**  $\triangleq$  Algorithm presented as pseudocode

**Code**  $\triangleq$  executable code to show the algorithm explicitly

**Table**  $\triangleq$  Explanatory information displayed in a table

**Animation**  $\triangleq$  a gif or animation of any type is used.

**Picture**  $\triangleq$  A photo/screenshot or picture is used.

**Sequence**  $\triangleq$  A conjunction of steps meant to show progress in a serial manner

**Plot**  $\triangleq$  A mathematically generated plot that adheres to some coordinate system

for example a definition might be described in English, followed by the same definition described by geometry. Notations can be combined for a single location like so:

$$\frac{n \in \text{Notation} \quad m \in \text{Notation}}{n/m \in \text{Notation}} \quad (1)$$