Analysis of Execution Trace Gantt Charts Experiment Documentation

Connor Scully-Allison & Kate Isaacs

1 Abstract

This document presents the various details and technical information required to understand this study and the code which realizes it. The study itself is an exploration into how well people are able to recognize and identify patterns of repeating lines connecting rectangular blocks. In this document we propose the background, general design and hypotheses of this study in addition to some technical details required for the maintenance and execution of code required for this study.

2 Core Research Goal

Understand what information *must* be included and can be omitted in dense Gantt charts conveying message passing information with lines. We want to understand how we can simplify these charts and how people perceive patterns.

3 Background

In the world of parallel computing, Gantt charts are used to show the execution of processes over time. They can be used to trace execution of parallel processes and monitor where and when certain communications take place between CPUs. Traditionally they have been used as part of a suite of other analysis tools [4, 3]; however, they are very powerful visualizations and can provide significant insight into the execution of parallel processes and especially if communication patterns conform to visual patterns that indicate a particular message passing structure.

3.1 Problem

Despite the insight Gantt charts *can* provide, as computing resources are scaling up, the effectiveness of these charts are not keeping pace. When the number of cores being analyzed is low (8-32), it is possible to trace lines between processes and extract meaningful information from this chart at a glance, an example of this is Fig. 1. However as the number of processes scale from the tens to the thousands the visualization becomes incomprehensible, as can be seen if Fig. 2.

4 Study

4.1 Study Purpose

Following from the problem in 3.1, a gap exists in our understanding of how to construct a Gantt chart that scales effectively with an increasing number of processes. In order to better understand how to construct such a chart, we must first get a more fundamental understanding of how people recognize patterns of repeating lines with incomplete or partial information.

Being able to recognize these patterns even with incomplete information is crucial to understanding what information can be safely omitted from visualizations in general or high level views. It also provides insight into what details are necessary so that a visualization does not become meaningless in general views.

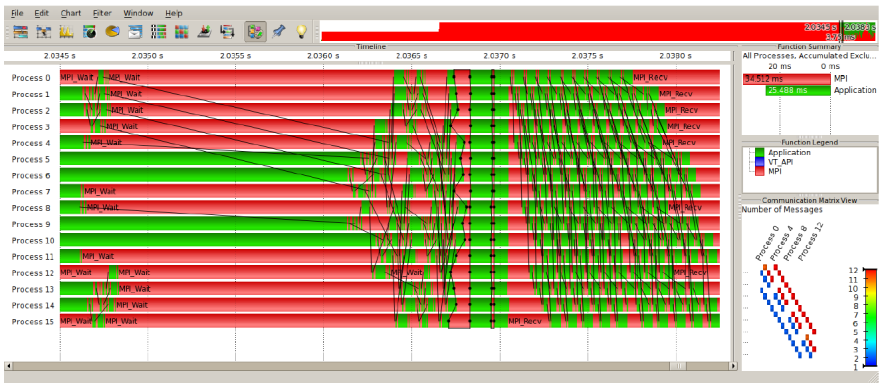
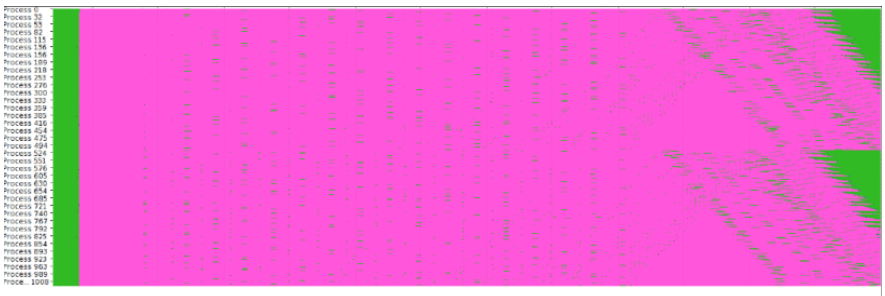


Figure 1: A Gantt chart produced by Vampir from [2]. With this number of concurrent processes, patterns can be seen in the lines denoting mpi calls. A "ring" can be seen between the first four processes and subsequent four. Even at this resolution, however, it can be hard to understand what is going on exactly on the right hand side.



(a) Vampir visualization of all 1,024 processes.

Figure 2: A Gantt chart with over 1000 processes. Image from [1].

4.2 Hypotheses

Some possible hypotheses:

1. Subjects will be able to relate patterns when going from full representations to full representations
2. Patterns with notable outlying features will be easier to identify and recognize like a ring or an offset pattern with multiple offset points.
3. (From pilot study) Subjects will prioritize angles over other pattern attributes when comparing patterns

4.3 Study Design

Subject Organization	Within-subjects	
Dependant Variable	Error Rate	
Independent Variable	Question Prompt Type	[Partial, Full]
Random Variable (A)	Call Trace Pattern	

Table 1: Components of study design.

Dependant Variable - Question Response

The dependant variable is a simple true/false value that indicates whether the respondent correctly identified

the presented "answer" pattern as matching or not matching the "question" pattern. This variable is categorical data and accordingly cannot be analyzed using an ANOVA analysis however it does possibly allow for the use of a χ^2 to show statistical significance of the influence of the following two independent variables on user responses.

Independent Variable A - Question Prompt Type

Our first independent variable "Question Type" describes the amount of information a subject is given for a question and it's associated answer choices. The "levels" of this variable are as follows:

1. Full Question — Full Answer
2. Partial Question — Full Answer

Random Variable - Call Trace Pattern

The second independent variable in this experiment is the type of pattern being rendered. These various patterns correspond to various Gantt chart visualization patterns which occur in real parallel processes. Presently there are four different levels for this variable:

1. Offset (Fig. 3a)
2. Ring (Fig. 3b)
3. Stencil (Fig. 3c)
4. Exchange (Fig. 3d)

4.4 Design Changes

1. Full and partial question types with only full answers be used to simplify the design of this experiment

5 Design Iterations

5.1 Open Interview Design

See pdf: Open_Interview_Report_Summary

5.2 Drawing Study

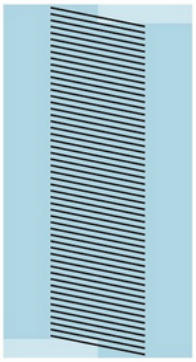
Abstract

In December 2019 and January 2020 this study was tweaked into a new pilot which explored the viability of giving a partial or full representation of a pattern of lines to a subject and asking them to extrapolate from it (if partial) or reduce it (if full). This study was executed on a cross section [c1] of attendee's to Hack Arizona – recruited with stuffed animal toys – totaling 15 respondents. After a semi-subjective grading process it was determined that nearly half of all responses had to be thrown out due to a misunderstanding of the question prompt(s). After this winnowing, there was not sufficient data remaining to make conclusions about how certain independant variables impacted dependant variables.

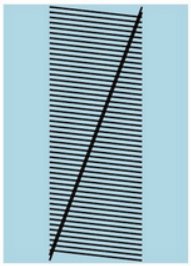
Technical & Study Design

For this study, our online platform was modified with the addition of a page which iterated through various generated patterns, placed them adjacent to an empty grid and formatted them in a way which made

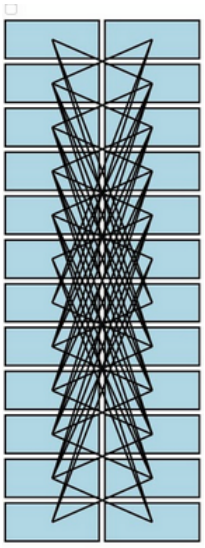
[c1] *kate*: Is this really random? I thought we recruited with plushies



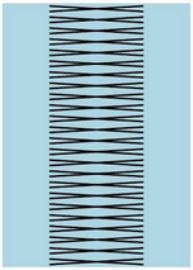
(a) An example of an offset pattern.



(b) An example of an ring pattern.



(c) An example of a stencil pattern.



(d) An example of a exchange pattern.

Figure 3: Examples of various execution trace patterns.

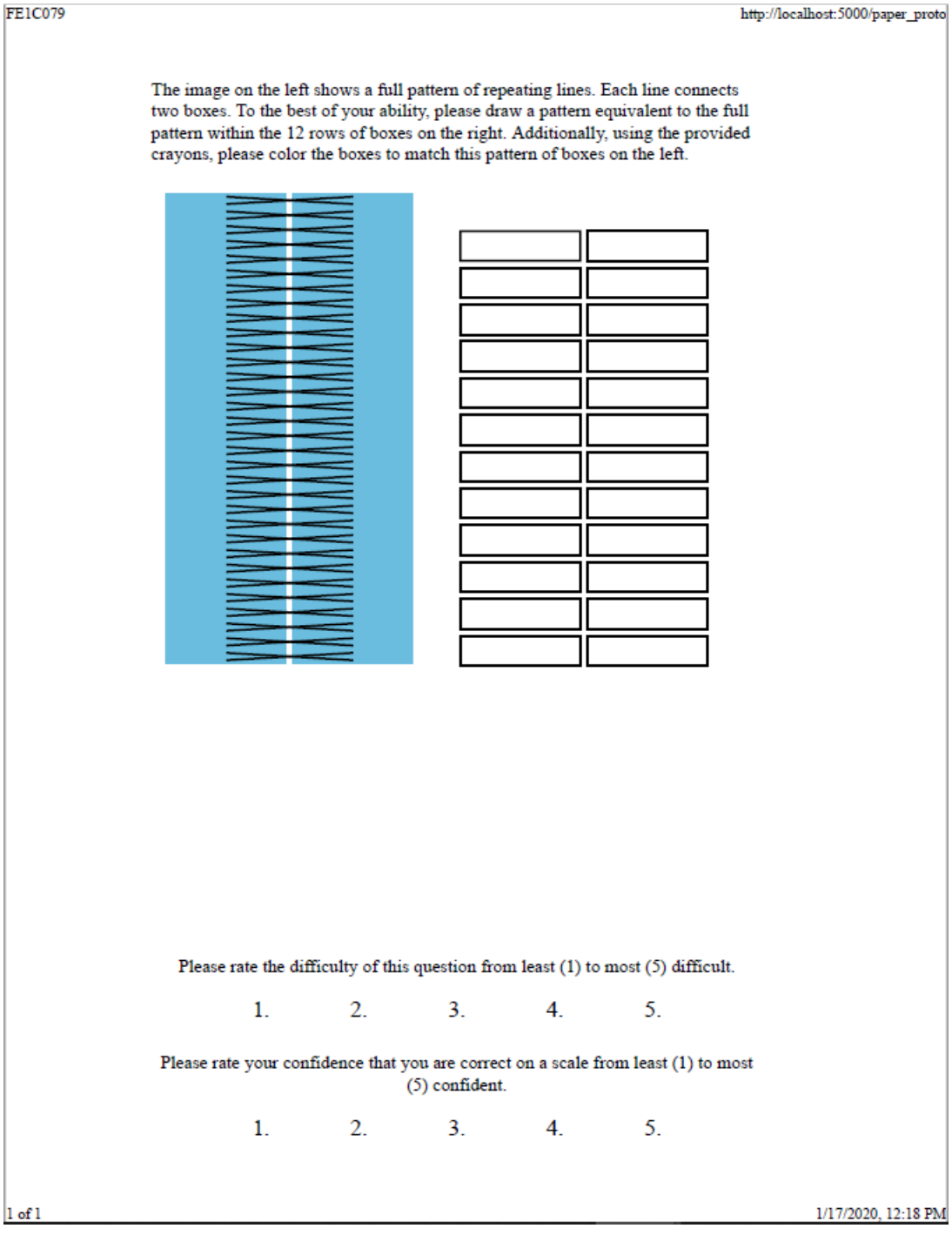


Figure 4: An example of the drawing experiment.

them easy to print to 8.5x11in paper. An example of this page can be seen in Figure 4. Approximately 20 packets containing four questions each we were produced with semi-randomized combinations of independent variables: Question Prompt Type, Call Trace Pattern (including grouped patterns), and Partial Extraction Location. The height of full representations was varied as well as a random variable, representing different "zoom levels" on full patterns.

6 Improved Visual Design

For the improved visual design of a Gantt chart which is ordered using logical time we are expecting to use a glyph which abstracts the underlying pattern and groupings in our chart and conveys them in a limited amount of space.

6.1 Encodings

This is a list of all the variables we are attempting to encode into our glyph.

- Base pattern
  - Offset
  - Ring
  - Exchange
- Location in Time
- Grouping Factor
- Continuation
- Scope of Icon Reference
- *Crossing Nodes (later)

6.2 Channels

Here we are documenting what channels we have considered and why or why not they may be a good option for use in our design/glyph. This breakdown can be seen in Table 2

6.3 Considerations

6.4 Proposed Encoding Rules

1. Pattern $->$ shape: The abstract shape of a pattern (comprised of a few parallel or overlapping lines) will convey this general pattern. This will encode the direction of a slope from right to left (downward, upward) and will also convey sharp outliers in slope patterns as with a ring.
2. Grouping Factor $->$ color & number & y-position: Grouping will be conveyed with a doubling up of a pattern and positioning one on top of the other. Then, we will render one pattern wholly in black and the other in blue.
3. Continuation $->$ opacity & y-position: Continuation of a pattern will be conveyed with a duplication of our core pattern, placing it in the direction of continuation and reducing the opacity of this pattern on a gradient.
4. Location in time $->$ x-Position: The logical-temporal occurrence of a particular pattern will be encoded by being placed over those time steps on the x position

Position on a Common Scale	We will likely place the entire glyph/icon on a common scale to convey that different patterns are associated with different columns. The common scale will be the regular time intervals denoted on the x axis. Untied to x values. c1
Position on an Unaligned Scale	The unaligned scale does not seem to meaningfully convey time of a glyph in the same way that an common scale does. ~~We could place the entire glyph down on an unaligned scale as well. It will show that two glyphs are distinct. This could only likely work for the x axis.~~
Length	We cannot directly encode line length as it is directly related to the distance between calls to locations and the time step between calls as well. c2
Tilt/Angle	Angle could be used to encode or convey or hint at the offset of our pattern. However, we have determined that offset is not a salient characteristic under many circumstances and will only be relevant when crossing nodes. ~~and angle will also be used to encode a particular pattern in general. As exchanges and rings and offsets require differently angled lines.~~ c3
Area	Area could be used to convey that a pattern applies to a pair of processes at logical timesteps. It would encode that the patterns are referring to the horizontal and vertical space between t and t-1.
Depth	We do not think depth will be a very helpful encoding attribute at this time. For our 2d focused designs it would be expected to complicate the design without adding very much.
Luminance/Saturation/Opacity	Opacity could be used to convey boolean data of continuation off the edge of the drawing area. We can duplicate the patterns we have already drawn and render at a lower opacity them above or below our pattern, where they continue.
Curvature	Curvature is not needed to convey any information in our dataset. At our current iteration we are using only straight lines and rectangular boxes.
Volume	Our current data is plotted along two axes right now and we do not expect that encoding some value in 3D will convey information which he have not conveyed elsewhere.
Spatial Region	Spatial region could be used to convey the scope of a certain pattern in a way similar to area.
Color Hue	Using the gestalt principal of similarity we will use color to convey groupings of lines in a repeating pattern. We will alternate colors over the same visual pattern to convey groups within a particular grouped pattern.
Motion	Motion will likely not be used because it is a very strong popout channel and will be too distracting.
Shape	Shape will not encode any information directly. If we conveyed patterns as shapes we anticipate that we will diverge too far from what our user base expects from a Gantt chart visualization. c4
Closure	We may enclose each of our pattern glyphs in a box or icon to draw the attention of users to that portion of the chart.
Density/Contrast	Density could be used to convey (very roughly) how many lines are contained within a group. This does pose problems of making the pattern unrecognizable with groupings of many-many lines.
Number	Like density, the number of parallel lines in a particular pattern could convey or hint at the total number of lines the underlying pattern has. Perhaps with a mapping of 1 line = 1000 real lines. Number could be used to convey grouping by doubling up a base pattern.
Orientation	Since angle will be used to convey to the user what pattern our software is encoding, mapping orientation to some variables seems like it would only make the visualization more confusing.

Table 2: A table of possible channels to consider and reasons for accepting/ rejecting them

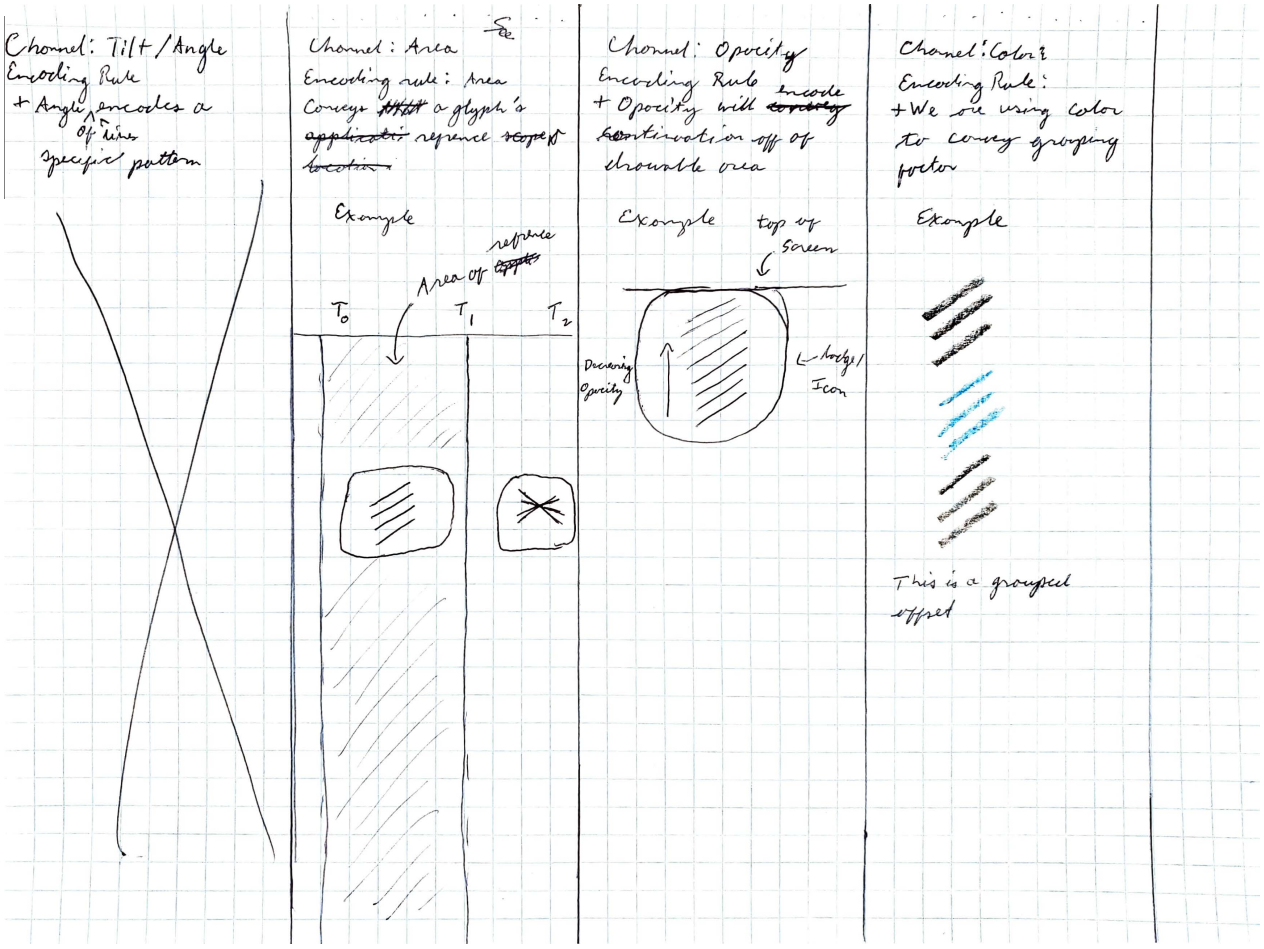


Figure 5: Rough visualizations of encoding rules.

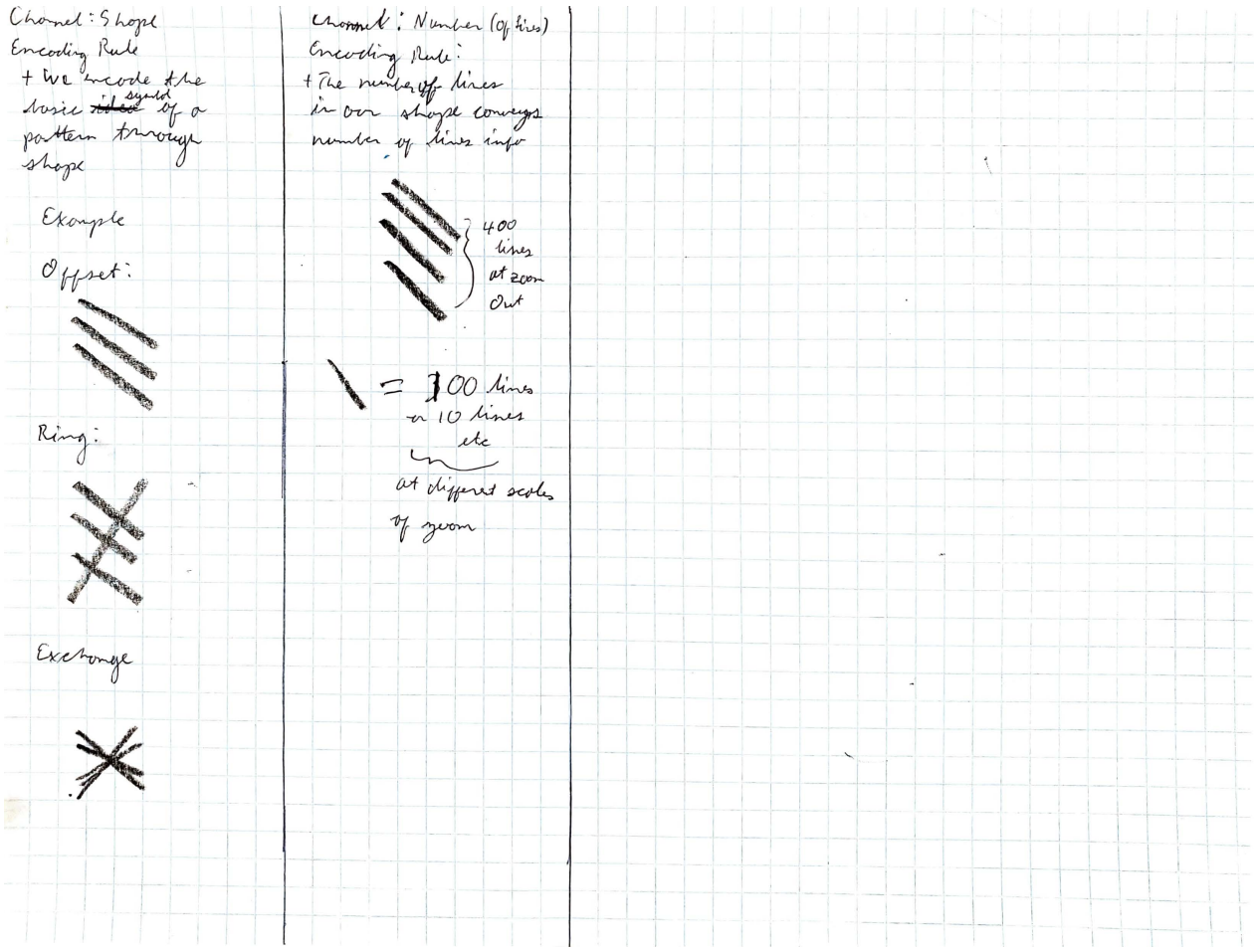


Figure 6: Rough visualizations of some additional encoding rules. Number of lines will not work as well for exchanges and rings. It will require additional info.

6.5 New Proposed Designs

From the above observations and discussions concerning what we should encode and how it should be encoded we arrived at the new proposed designs which can be seen in Figure 7.

In this figure, the three of the main patterns or "shapes" which naturally occur in logical-time Gantt charts are represented here: offset, rings and exchanges. These primitive abstractions of real communication patterns, can be overlayed on top of a Gantt chart when the resolution is too small to render real communication lines meaningfully. They will be placed along the x axis, aligned with the underlying patterns they represent.

For rings and offsets, differences in distance of communication are expressed with the use of angle. The angle of the "horizontal" lines increases from 15 degrees to 60 degrees. They do not encode the true distance of communication between nodes, but rather hint at the magnitude of distance over which communication is occurring. For exchanges we abstract the idea of distance further by showing farther exchanges as increasingly dense tighter angled lines. To preserve space however, the mappings are not direct to the degrees used in the prior primitives.

Although not pictured here, a gradient can be placed on top of these patterns to represent that they continue past the bounds of the currently rendered area. These designs have the additional benefit of being vertically collapsed and expanded by removing intermediary "horizontal lines." The visual structure of each of these figures remains clear when represented with as few as three horizontal lines. This is beneficial because it allows us to represent groups of patterns using the same amount of space as one long pattern, while persevering individual characteristics like distance and structure in a recognizable way.

Finally, from a visual design perspective, we opted to render the paths of our "communication lines" with narrow centers and tapered ends to distinctly show, at a glance, that they are not the real lines. We intend for this to prevent possible confusion because these representations use the same primitives of lines to show communication but have different meaning from those normally used in Gantt charts. The specific style chosen was made to emulate an engineer's pen which would be used in drafting blueprints or making designs.

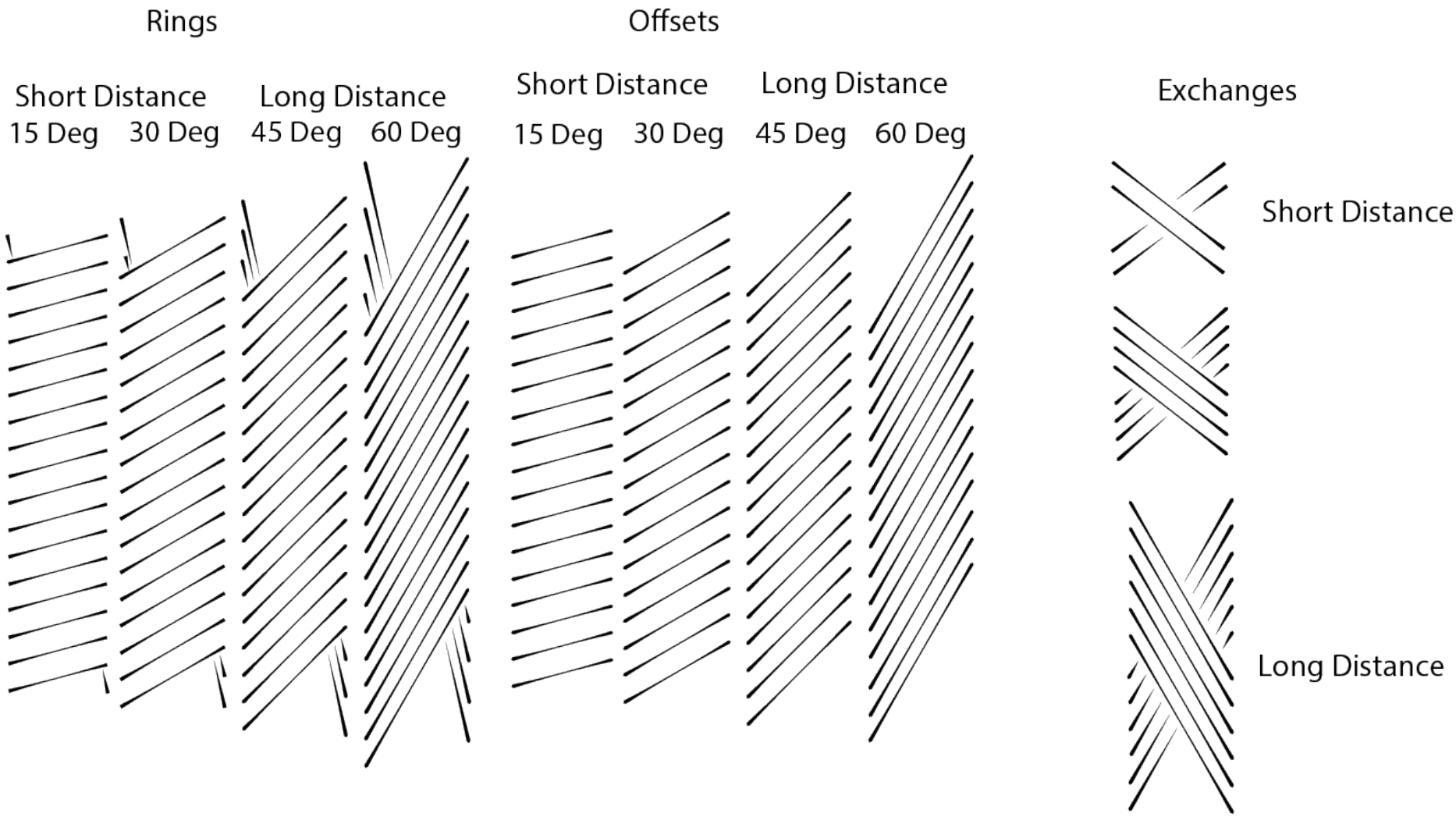


Figure 7: A breakdown of the slightly polished designs we propose as an alternative to the visual noise generated by "zoomed out" logical-time Gantt charts.

7 New Evaluation Plans

As our experiment has undergone many iterations we were able to propose a new and novel design for conveying the general structure of a communication of a Gantt chart at a high level. The proposed designs can be seen in Figure n. With the advent of these new designs however; out prior experiment no long seems to meaningfully test whether our new design will be better or not. So we must propose a new test, one that appropriately tests the metrics which are most meaningful to users of Gantt charts.

7.1 New Test Preliminary Design Considerations

8 Code Documentation

8.1 Tools and Technologies

The software for this experiment is coded using Python w/ Flask to serve down web pages and record results on the host computer. On the front-end, Javascript with D3 is used to handle the drawing of patterns and the collection of data from user inputs.

8.2 File Organization

From the top level of the source folder you will see four subfolders:

- data/
- parseScript/
- static/
- templates/

The bottom two folders contain all necessary source files to run this experiment. *templates/* contains HTML templates served down with the python flask service *study.py*. *static/* contains all other static files required by the html templates. This includes all images, css files and the core Javascript routines which handle the rendering of pattern charts.

The top folder, *data/*, contains data collected from prior experiments in addition to the answer keys for all questions that have been used in prior experiments. Parse and data-analysis scripts are located in the *Old Data/* subfolder. They are written in Python.

The *parseScript/* folder contains a parse script written by Sarah for an earlier iteration of this experiment and analysis. It has been kept for posterity, but may be removed later.

References

[1] Katherine E Isaacs, Peer-Timo Bremer, Ilir Jusufi, Todd Gamblin, Abhinav Bhatele, Martin Schulz, and Bernd Hamann. Combing the communication hairball: Visualizing parallel execution traces using logical time. *IEEE transactions on visualization and computer graphics*, 20(12):2349–2358, 2014.

[2] Katherine E Isaacs, Alfredo Giménez, Ilir Jusufi, Todd Gamblin, Abhinav Bhatele, Martin Schulz, Bernd Hamann, and Peer-Timo Bremer. State of the art of performance visualization. In *EuroVis (STARs)*, 2014.

[3] Thomas J LeBlanc, John M Mellor-Crummey, and Robert J Fowler. Analyzing parallel program executions using multiple views. *Journal of Parallel and Distributed Computing*, 9(2):203–217, 1990.

[4] Wolfgang E Nagel, Alfred Arnold, Michael Weber, Hans-Christian Hoppe, and Karl Solchenbach. Vampir: Visualization and analysis of mpi resources. *Supercomputer*, 1996.