BUSII
Assignment 1: Data Understanding

Sonja Biedermann Kaur Karus Christian Luidold

October 24, 2018

1 Observations

1.1 Tree Structure

We decided to start our preliminary analysis of the log files by generating a tree of the general process. Theoretically, there should be one root process which spawns a production process for every one of the five parts, which in turns spawns four processes—two each for turning and milling—which then execute collection and control workflows.

The general procedure seems to be a call to an URL ending in `/flow/start`. After some time, a message with the instance ID is received by the parent process. The process is slightly different for the root, which calls the same URL, but gets back another URL which contains the ID (instead of just getting the ID).

Unfortunately, the full tree cannot be recovered this way. What we get is a partial tree, and additionally (when forcing our script to explore all log files, and not just the ones it found references too), a few partial trees and lone orphan nodes to which no references can be found.

We have also noticed that the root, firstly, only explicitly gets the URL of 4 started processes, and secondly, that one of these 4 goes by the ID 180, which cannot be found in the data set. It seems that something went wrong with that one. After looking at the other processes, we found two process instances which are labeled as “Plain Instances” which from their looks seem to be the two missing production processes.

Notably, most all partial and especially the orphaned processes appear to be milling instances. We provide the generated forest in Figure 1. As you can see, the figure is rather wide. We provide the pdf in the supplementary zip file if you want to take a closer look.

Processes related to turning are colored blue (different shades depending on production or machining), and milling related processes have red to brownish colors. Yellow is used for the plain instances, which we are not sure about but do believe to be just the missing production processes. Note that we expect 2 turning processes to be started, but they only start the second turning process. There are also no orphaned turn-1 processes, so it appears that they were skipped.

We’ve also included the part numbers in the nodes. It is clear that the part numbers do not always match, e.g. children of some process allegedly belonging to part 3 are assigned

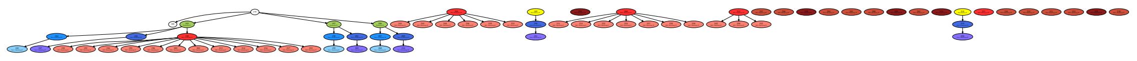


Figure 1: Forest resulting from an exhaustive depth-first search of all logs

to another part, or children are assigned to different parts altogether. This does not seem to be too grave since the part label appears to be mostly informational. But it does show that something is wrong.

Overall, it appears that most processes we expected (except for the turning 1 subprocesses of the plain instances, as mentioned) are indeed there, just not explicitly linked by the logs. We attempted to find references to some instances by using good old `grep`, e.g. by doing `grep -R "'297'"`, with 297 being one of the orphan milling processes, and were unable to find any mention of this ID which is not in the process' own log file.

This begs the question on how to piece together this forest into a single tree if we cannot trust the part labeling. We could use the timestamps for further context—after all, when a parent is spawning its children in one go, they are probably temporally close to each other. However, since everything is running potentially concurrently, there seems to be a big potential for ambiguity.

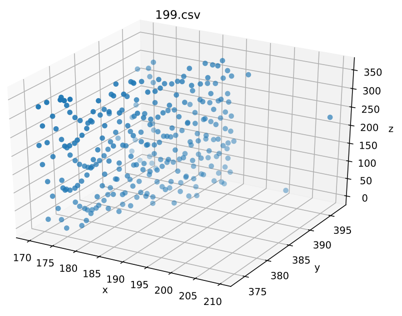
1.2 Coordinates

But what do the processes *do*?! To get some insight on this, we extracted all the coordinates that are reported to the process. We picked a few promising ones for Figure 2. We have to admit that we have a bit of a stupid approach to the timing of the reported coordinates, namely we order them by timestamp and then just take every (x, y, z) tuple as-is. This cannot represent moves where e.g. one or both coordinates stayed constant. Something smarter should be implemented—we should try to use the values that are the most recent to the other coordinates. There is, however some structure to the generated scatterplots, which makes us hopeful that a better approach would yield even prettier (or the same) pictures. Now if we only knew more about gas turbines.

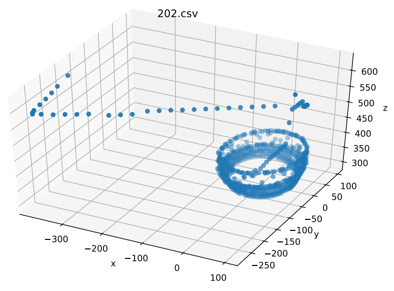
The milling processes clearly show that something circular is happening, which makes sense since the lowerhousing *does* appear to be circular. There also seems to be some milling going upwards (in z direction). The turning processes seem to do wavy motions. A next step of this would be to analyze where errors occur, e.g. was the trailing up and off in subfigure (b) initialization of the machine or did the motion occur due to human intervention (due to error).

1.3 Process Templates and Versions

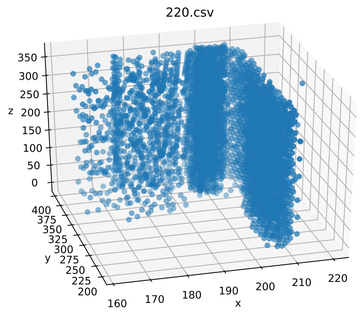
Concerning matching the logs to the various process templates the majority of the logs correspond to a single version of their given process, with the exception of the logs with the ID of 329 and above, although the processes concerning "Machining V2" could also be assigned to "Machining V1" as the only difference seems to be the lack of the activity "Status" as well as a looping functionality. Considering the IDs 329 and 334 can definitely be assigned to "Production V2", as they differ considerably from "Production V1", it's safe to assume the IDs in question belong to "Machining V2". The following list shows the IDs of the logs matching their corresponding processes split according to their given part:



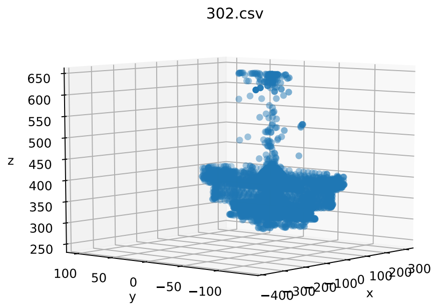
(a) Process 199, Turning 1



(b) Process 202, Milling 1



(c) Process 220, Turning 2



(d) Process 302, Milling 2

Figure 2: A selection of machining patterns

- **Lowerhousing Order Processing**
  - **root:** 179
- **Lowerhousing Production V1**
- **Lowerhousing Production V2**
  - **part1:** 183[0]
  - **part2:** 189[0]
  - **part3:** 197[0]
  - **part4:** 206[0]
  - **part5:** 218[0]
- **Lowerhousing Production V3**
- **Machining V1**
  - **part1:** 185, 187, 192, 193, 194, 195, 196, 202, 205, 211, 213, 214, 215, 216, 217, 294

[0] As Lowerhousing Production V2 & V3 seem to be equal they can be assigned to both of them

  - **part2:** 191, 237, 238, 239, 240, 242, 243, 244, 245, 246, 296, 297
  - **part3:** 199, 203, 250, 300, 301, 302
  - **part4:** 208, 210, 252
  - **part5:** 220, 222, 223, 224, 225
- **Machining V2**
  - **part4:** 330, 331, 332, 333
  - **part5:** 336
- **Production V1**
  - **part1:** 184, 186, 188, 226
  - **part2:** 190, 200, 295
  - **part3:** 198, 201, 204, 299
  - **part4:** 207, 209, 212
  - **part5:** 219, 221
- **Production V2**
  - **part4:** 329
  - **part5:** 334

1.4 Process Evolution

The main differences between V1 and V2 seem to be additional activities regarding the status-check most likely concerning quality control and/ or safety purposes. Furthermore, considering most of the provided logs are part of the processes "Machining V1" and "Production V1" we can deduce, that V1 established itself as the dominant strategy.

Concerning "Lowerhousing Production V2" and "Lowerhousing Production V3" they seem to be the collapsed versions of "Lowerhousing Production V1" in terms of functionality. Given the provided logs the segmentation of the different production processes became the dominant strategy.

2 Questions

What is the relevance of the `ClientHandle` value that appears often in Machining processes? It appears like an ID, but of what? Is it related to the programs running the machining or . . . ?

How to proceed with piecing together the tree, e.g. what precisely does the missing ID 180 mean? We assume that this means the process just died pretty early on or was cancelled by a human for some reason (e.g. so Turn 1 machining is skipped).

What's up with those orphan nodes, i.e. all the processes which we cannot find a spawn record for in the logs? Were they started by humans? Why are they different? Are the logs incomplete or did the instances just not wait for the receipt of the IDs of the spawned instances?

Can we just reorder the logs in the directories such that the part labels in the tree match? We can't really decide which part some subtrees address, e.g. some partial trees have a healthy mix of part 2, 3, and 5 in about equal amounts. So we need to pick one, and this then might not match the actual part number that was produced in real life. Is this important?

3 Possible Problem Statements

From our point of view one interesting question would be to find out where and how issues arise in the machining processes. One approach could be to visualize the path the drill (or whatever) took and then mark the location where the process stopped, presuming that the stop occurred due to error. However the usefulness of this is questionable, since there appear to be many human quality control measures, which probably already yield all the insights that an analysis of the logs could yield.

Outlier analysis of some of the values produced by the machining tasks could yield some information about weird values and what effect they have (e.g., resulting in an error). This could for example result in models that can automatically stop the machining process or alert a human which may be more effective.

We cannot think of other useful problem settings (i.e. suited for a larger project), which may be partly due to us not really knowing what would be useful due to lack of domain knowledge.

4 Conclusions

Working with this data sure seems like a challenge.