3.7 Iteration 4: Expanding data

We noticed that the architectural issues all emails pairs consistently outperformed the architectural emails all issues pairs. We hypothesized that this difference was because we have a architectural email dataset of only 620 emails from 128 different email threads. Meanwhile we have 1426 architectural issues. We wanted to test if increasing the architectural data set would increase precision. In order to do this we were able to find a bigger data set of the same issues but with more labeled, with a total of 2166 laballed as architectural compared to our previous 1426. We were however not able to find a better data set for architectural labelled emails. This meant that we could only test this hypothesis in the ArchIssuesAllEmails pairs. We then needed to decide which method we use to calculate our similarity values. We decided to use the model averaging of iteration 3 because we noticed that it is more precise in finding ADDIE-pairs in the first 20 pairs, but dropping afterwards. We expect that when we increase the data that better precision will continue for longer. The results are shown in table 10. We see a 34.78% increase in ADDIE-pairs when comparing iteration 4

	Iteration 3	Iteration 4
Number of ADDIE-pairs	46	62
Number of unrelated pairs	54	38

Table 10: Results from the first 100 pairs in Iteration 4 (Increasing architectural issue data) compared with iteration 3

with iteration 3, supporting our hypothesis that increasing the architectural data increases precision. We decided to analyze a total of 200 pairs, where we concluded that theoretical saturation was reached. This conclusion was reached based on the fact that we began to find very little ADDIE-pairs in the last 50 pairs. In total we found 82 ADDIE-pairs in the top 200 pairs, only finding 20 new ADDIE-pairs in the last 100. During this iteration we also refined our ADDIE-pair patterns, which we discuss in 4.1.

3.8 Resulting Dataset

After these 5 iterations we have a total of 26 tables, which can be found in appendix D. Of these tables 9 contain the data used to analyze each iteration, 2 contain the all analyzed pairs (with the difference being how the ADD types are determined), 6 contain the all similarity pairs with a cosine similarity greater than 0 (3 for SBERT and 3 for TF-IDF), 3 containing the email and jira data and 6 containing extended data which is used to prepare for later iterations. For future work the table unique_pairs could be the most beneficial since it contains 681 unique pairs judged if they are both about the same ADD and what ADD types the issue is.

4 Results

4.1 RQ1: Issue - email thread patterns about the same ADD

During each iteration we created patterns which ADDIE-pairs seem to follow and assigned them to each ADDIE-pair. We analyzed a total of 681 unique pairs, 163 of them were ADDIE-pairs, 432 were unrelated and 86 architecturally irrelevant. We assigned each ADDIE-pair a pattern during each iteration, refining the patterns after each iteration. We reached theoretical saturation in iteration 4. We got in total 6 patterns based on the total of 163 unique ADDIE-pairs. Table 11 shows how many times each pattern did occur.

4.1.1 Pattern 1: Initiate ADD in Email Thread and Discuss in Issue.

In this pattern there is a begin of the ADD in the email (thread), which is almost directly moved to an issue. The email (thread) contains either an ADD or a plan for an ADD. There is the option for a short architectural relevant discussion to take place in the email thread, but this is the exception rather than the rule. The initial email asks for feedback about its ADD proposal (e.g. "What do you think?" or "Is this something we need?"). Often there are simple reply emails that voice their approval of the proposal. An issue will be created shortly (in less than a week) afterwards, and often but not always a reply with an url

Pattern	Occurrences
1	19
2	33
3	26
4	25
5	49
6	11

Table 11: ADDIE-pair patterns and their occurrences

(or JIRA key) to an issue will be send. In this issue there exists architectural relevant information or an architectural discussion in the issue comments.

Example	email id = 34469 [5]	issue key = CASSANDRA-14448 [6]
Source	Rational of discussion	quote
email	Initializing ADD	"I'm working on some performance improvements of the lightweight transitions..." "...current CAS requires 4 round trips to finish ..." "I'm proposing the following improvements to reduce it to 2 round trips."
Email	Requesting action / feedback	"What do you think? Did I miss anything?"
Jira	Creating Jira and adding ADD in description	The Jira contains (nearly) the same text as the initial email.
email	Notifying about move to Jira	"Cool, create a jira for it, https://issues.apache.org/jira/browse/CASSANDRA-14448."
email	Discussing about (architectural) information in email thread	"... if we combine the prepare and quorum read to- gether... We can improve it by avoid executing the read, if the replica already promised a ballot great than the prepared one.
Jira	Discussing about (architectural) information in comments	Person responding to 'Combine prepare and quorum read together' with "It's a tradeoff though, not a clear cut optimization: it will certainly speed up the non- contended case, but every time the prepare fails, you will have wasted time/resources on some reads..."

Table 12: Example of pattern 1, with source being in chronological order

4.1.2 Pattern 2: ADD in issue with Email as Feedback Request

This pattern is almost entirely based in the issue. Only at one point in time there is an email sent out to get the recipients to take action, often in the request of feedback. It can be that an issue has been created but no activity has been shown. An email will be sent out with architectural relevant information (sometimes the description is copy pasted) in order to gain attention.

[5] email url = https://www.mail-archive.com/dev@cassandra.apache.org/msg12600.html
[6] issue url = https://issues.apache.org/jira/browse/CASSANDRA-14448

Example	email id = 14279 [7]	issue key = HADOOP-8803 [8]
Source	Rational of discussion	quote
Jira	Reason for creating ADD	"I am modifying the Hadoop's code ... to achieve better security."
Jira	Requesting action / feedback	"I want to know that whether community is interesting about my work? Is that a value work to contribute to production Hadoop?"
email	Notifying about existence of Jira	The same text as the Jira issue description, together with "I created JIRA for the discussion. https://issues.apache.org/jira/browse/HADOOP-8803#comment-13455025 "
Jira	(Optional) Discussing about (architectural) information in Jira issue comments	"Makes sense, but I think it's going to be difficult to plumb through the various abstractions here in a clean way that doesn't introduce specific dependencies on FileInputFormat, etc."

Table 13: Example of pattern 2, with source being in chronological order

4.1.3 Pattern 3: Create ADD in Email Thread and Implement in Issue

This pattern has the architectural design and discussion in the email thread and the implementation in the Jira issue. It often starts as a proposal or request for an ADD. Questions about the ADD are asked or answered and the ADD will be refined. Eventually a Jira issue will be created in order to begin working on the implementation of the ADD. There are also cases were an a person wants to submit an ADD to a project and sends an email about it. We include those emails also in this pattern because the issue is mostly about implementation and not about changing the ADD.

Example	email id = 8561 [9]	issue key = CASSANDRA-2017 [10]
Source	Rational of discussion	quote
Email	Initiating creation of ADD and requesting discussion	[Replacing ivy with maven-ant-tasks] "Is this something that people are OK with? It will result in the version details being specified from the build.xml and not a separate ivy.xml"
Email	Discussing about ADD	"Why? What are the advantages?" "1. It will make deploying to central easier ... 2. You seemed to think it would be better keeping all the version information in build.xml rather than in a separate file 3. ..."
Email	Requesting move to Jira / requesting start of implementation	"If everyone is OK I'll create a JIRA against Core with fixVersion 0.7 and add my patch there." NOTE: this was the first email but only after the last email in the thread was sent, was the JIRA issue created
Jira	Creating Jira containing ADD	"Replace ivy with maven-ant-tasks. Three main reasons: 1. In order to deploy cassandra to maven central, we will need to use maven-ant-tasks anyway 2. ... 3. ...
Jira	(Optional) Discussing about implementation of ADD in Jira issue comments	"Tested that ant release still works after realclean. On a standard build, dependency checking is dramatically faster than ivy's."

Table 14: Example of pattern 3, with source being in chronological order

[7] email url = https://www.mail-archive.com/common-dev@hadoop.apache.org/msg07358.html
[8] issue = https://issues.apache.org/jira/browse/HADOOP-8803
[9] email url = https://www.mail-archive.com/dev@cassandra.apache.org/msg01591.html
[10] issue = https://issues.apache.org/jira/browse/CASSANDRA-2017

4.1.4 Pattern 4: Release group

For pattern 4 we take a similar definition as W. Meijer's Release Group[12]. Here the email thread talks about the decision about releasing one or multiple issues (for example "symlink support in Hadoop 2 GA"[11]). Sometimes the reply emails can contain insightful discussion, sometimes a reply email consist of simply a vote (e.g. "+1"). Overall most if not all of the release discussion takes place inside the email thread, whereas the issue contains mostly information about an ADD. A difference between pattern 4 and the other patterns is that pattern 4 is less about the design of the ADD and is more focused on its release.

Example	email id = 28314 [12]	issue key = YARN-4356 [13]
Source	Rational of discussion	quote
Issue	Issue to be released	"*ensure the timeline service v.2 is disabled cleanly and has no impact when it's turned off*"
Email	Creating release discussion	"*I'd like to open a discussion on merging the Timeline Service v.2 feature to trunk (YARN-2928 and MAPREDUCE-6331) [1][2].*"
Email	Discussing about release	"*Big +1 on merging ATS-v2 to trunk. However, my concern to release it in 3.0.0-alpha (even as an alpha feature) is we haven't provide any security support in ATS v2 yet. Enabling this feature without understanding the risk here could be a disaster to end-user (even in a test cluster).*"
Email	Discussion reply	"*You're right. Can we document and clarify that it's still "alpha 1", and it doesn't have security features. I also think ATS 1.5 supports security features, so it's good for production - we should document it officially.*"
Email	"Conclusion"	"*Thanks everyone for chiming in on the discussion. Since no blockers were raised, I'll go ahead and start a vote thread.*"

Table 15: Example of pattern 4, with source being in chronological order

4.1.5 Pattern 5: Feature group

For pattern 5 we found that these pairs are very similar to the Feature Group defined by W. Meijer[12]. The ADDIE-pairs are less following a strict pattern and are more defined by other characteristics. Pattern 5 is assigned to ADDIE-pairs where the email thread discusses an ADD that contains other smaller ADDS. An example of such an email thread is the email titled "[DISCUSS] Hadoop SSO/Token Server Components"[14], where they list 8 required sub-components for their main component (that being the Hadoop SSO/Token Server Components). These sub-components can be seen as their own ADDS, resulting in this email being related to multiple ADDS. This could be seen as a collection of pattern 4s. This makes these email threads very big collections of architectural knowledge, but it is harder to link them to a specific issue, with the exception of umbrella issues. We assign this pattern to ADDIE-pairs if: 1) The email thread contains discussions about multiple ADDS; Or 2) The issue is a subtask of an issue which is related to an email thread like situation 1; Or 3) The issue is a collection of subtasks that are smaller ADDs; Or 4) The issue is an implementation of the ADD discussed in an email thread that discusses multiple ADD issues.
The issue of the pair from this pattern often contain little architectural discussion since the discussion already took place in the big email thread. Unlike other patterns which follow a pattern with their emails and issues, with pattern 5 we don't know anything about the chronological appearances of either the email thread or issue. Therefore the example in table 16 could differ from other cases. In our issue data set, we saw that (HADOOP) 'Project Rhino' and 'Cassandra Enhance Proposal (CEP)'[20] are most common in this pattern.

[11]email url = https://www.mail-archive.com/common-dev@hadoop.apache.org/msg10597.html
[12]email url = https://www.mail-archive.com/yarn-dev@hadoop.apache.org/msg23808.html
[13]issue url = https://issues.apache.org/jira/browse/YARN-4356
[14]email url = https://www.mail-archive.com/common-dev@hadoop.apache.org/msg09911.html

Example	email id = 17585 [15]	issue key = HADOOP-9392 [16]
Source	Rational of discussion	quote
Issue	Umbrella issue created	"*This is an umbrella entry for one of project Rhino's topic, for details of project Rhino, please refer to https://github.com/intel-hadoop/project-rhino/.*"
Email	Discussion email thread created	"*As a follow up to the discussions that were had during Hadoop Summit, I would like to introduce the discussion topic around the moving parts of a Hadoop SSO/Token Service. There are a couple of related Jira's that can be referenced and may or may not be updated as a result of this discuss thread.*"
Email	Listing of ADDS	"*Considering the above set of goals and high level interaction flow description, we can start to discuss the component inventory required to accomplish this vision: 1. SSO Server Instance:*" In total 8 componments are listen, which can be counted as ADDS
Email	Developing ADD	"*I have also updated our TokenAuth design in HADOOP-9392. The new revision incorporates feedback and suggestions in related discussion with the community, particularly from Microsoft and others attending the Security design lounge session at the Hadoop summit. Summary of the changes: 1. Revised the approach to now use two tokens, Identity Token plus Access Token, particularly considering our authorization framework and compatibility with HSSO;*" With more changes in the email.
Email	Discussion / conflict	Person is replying to a quote: "*"Personally, I think that continuing the separation of 9533 and 9392 will do this effort a disservice. There doesn't seem to be enough differences between the two to justify separate jiras anymore." Actually I see many key differences between 9392 and 9533. Andrew and Kai has also pointed out there are key differences when comparing 9392 and 9533. Please review the design doc we have uploaded to understand the differences.*"
Email	Discussion about subtasks	"*The following JIRA was filed to provide a token and basic authority implementation for this effort: https://issues.apache.org/jira/browse/HADOOP-9781*"

Table 16: Example of pattern 5, with source being in chronological order

4.1.6 Pattern 6: Initiate ADD in Issue and Discuss in Email Thread

This pattern is a mirrored version of pattern 1. Here an issue is created containing an initial ADD or idea, which is followed up by a discussion email thread. Often such a discussion would take place in the issue comments, but in order to get more attention an email is sent out. It therefore has a similar email intention to pattern 2, but where pattern 2 has the architectural relevant discussions in the issue comments, pattern 6 has them in the email thread. Optionally this pattern has an architectural discussion in the issue comments, but this is not always the case.

[15] email url = https://www.mail-archive.com/common-dev@hadoop.apache.org/msg09911.html
[16] issue url = https://issues.apache.org/jira/browse/HADOOP-9392

Example	email id = 18812 [17]	issue key = HDFS-5333[18]
Source	Rational of discussion	quote
Issue	Issue created containing ADD	"Issue description = *This is an umbrella jira for improving the current JSP-based HDFS Web UI."* and comment posted half an hour later by issue creator = "One task of this jira is to modernize the UIs, however, the most important task I want to achieve in this jira is to move from server-side JSP pages towards client-based, AJAX-styled HTML 5 web pages."
Email	Creation of email containing information about ADD	"*Jing Zhao and I recently have reimplemented the JSP-based web UIs in HTML 5 applications*" ... "*The abstractions between the UI and the core server are well-defined, decoupling the UI and the core hadoop servers.*"
Email	Requesting action / feedback	"*Your feedbacks are highly appreciated.*"
Email	Discussing about ADD	"*I have a few concerns about removing the old web UI, however: * If we're going to remove the old web UI, I think the new web UI has to have the same level of unit testing. We shouldn't go backwards in terms of unit testing. *"
Issue	(Optional) Discussion about ADD in issue comments	"I have concerns with this client-side js only approach, which is less secure than a progressively enhanced hybrid approach used by YARN. The recent gmail XSS fiasco highlights the issue."

Table 17: Example of pattern 6, with source being in chronological order

4.2 RQ2: Effective similarity methods of finding issue - email thread pairs that discuss the same ADD

Iteration	Summary
0	Context based similarity (TF-IDF)
1	Context based similarity (TF-IDF) + Filter
2	Semantic based similarity (SBERT) + Filter
3	Model averaging of iteration 1 and 2
4	Increasing data in iteration 3 (only ArchIssuesAllEmails)

Table 18: Definitions of the different iterations used

Iteration	ArchEmailAllIssue			ArchIssueAllEmail		
	unrelated	architecturally irrelevant	ADDIE	unrelated	architecturally irrelevant	ADDIE
0	45	17	38	43	16	41
1	39	16	45	34	16	50
2	61	3	56	50	8	42
3	46	8	46	43	12	45
4	–	–	–	23	15	62

Table 19: Results of the different iterations from the top 100 pairs in each table.

In order to answer the question of what similarity method would be best, we used 5 different ways of finding ADDIE-pairs with each method having a sample pool of 100 pairs. Figure 6 and 7 show the precision of each iteration, with figure 6 not having a graph for iteration 4 since we did not change anything for

[17] email url = https://www.mail-archive.com/hdfs-dev@hadoop.apache.org/msg11691.html
[18] issue = https://issues.apache.org/jira/browse/HDFS-5333

ArchEmailsAllIssues table during that iteration. We also put the numbers in table 19 where we split unrelated pairs into two categories. One category 'unrelated' contains the pairs that were not ADDIE-pairs but do contain architectural knowledge. The other category 'architecturally irrelevant' contains pairs where at least one element of the pair does not contain architectural knowledge. We see that the ArchIssuesAllEmails pairs outperforms the ArchEmailsAllIssues pairs. An explaination for this could be that we have more than twice as much architectural issues compared to emails. From these graphs we see that iteration 4 is clearly the most effective way of finding ADDIE-pairs. While we only apllied that method on the ArchIssuesAllEmails pairs we are confident that similar results could be obtainable when applied to ArchEmailsAllIssues.

We also saw in iteration 0 that with unfiltered TF-IDF cosine similarity, 91.26% of not related had an email or issue word count of less than 50. However if both email and issue have at least a word count of 50, then there is a 72.16% chance of the pair being an ADDIE-pair. Figure 4 shows us the frequency of the smallest word count on unrelated pairs and ADDIE-pairs, from which we can see that with a minimum word count of 50 (for both emails and issues) we can eliminate most unrelated pairs.

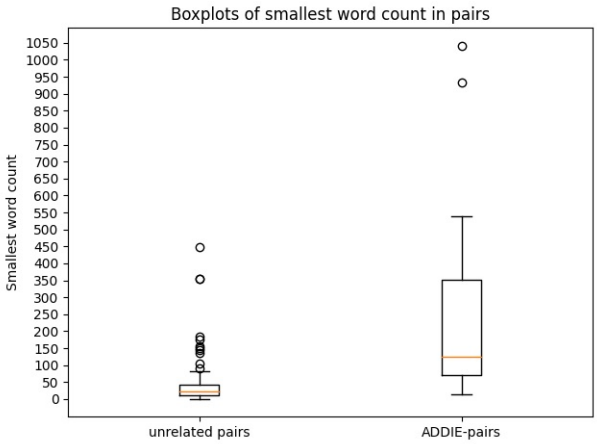

Figure 4: Boxplot of smallest word count in iteration 0 pairs, with ArchEmailsAllIssues and ArchIssuesAllEmails merged.

We also noticed that creation time difference played a big role for the effectiveness of similarity methods in finding issue - email thread pairs about the same ADD, with the ADDIE-pairs never having more than 700 days between the creation of the email and the issue. Here we take the creation time of each email and not just the creation time of the email thread. We determined that selecting pairs with a time difference of 500 or less between the creation of the email and the creation of the issue, that 57.35% of the pairs were ADDIE-pairs. Whilst we saw that if we took all pairs with a creation time difference greater than 500 that 98.44% of those pairs were unrelated. From this we conclude that with a maximum of 500 days between the creation of an email and issue we can eliminate most unrelated pairs.

For finding unique issue - email thread pairs that talk about the same ADD we a filter to reduce duplicates. For each email thread - parent issue combination, only taking the issue - email pair with the highest similarity score allowed us in iteration 0 to preemptively remove 12.5% of the pairs because they were duplicates. Although there are still duplicates, this filter will still increase the precision for finding unique issue - email thread pairs that talk about the same ADD.

When it came to comparing TF-IDF cosine similarity versus SBERT cosine we saw the following. When taking the top 100 pairs in both ArchEmailsAllIssues and ArchIssuesAllEmails tables and combining them, we saw that TF-IDF performed 21.79% better compared to SBERT. Despite having less precision, SBERT was able to find different ADDIE-pairs.
Taking the average of the two methods resulted in similar to precision as only using TF-IDF.

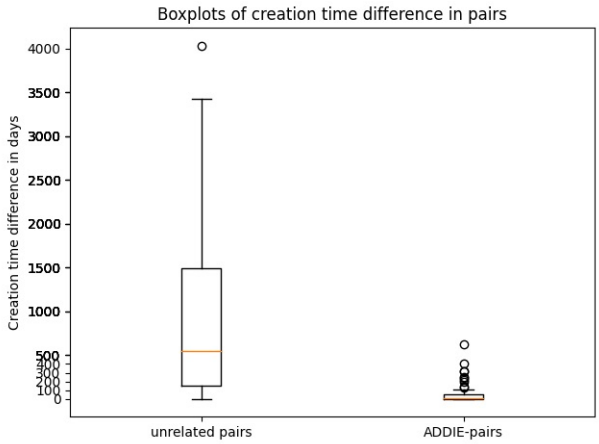

Figure 5: Boxplot of creation time difference in iteration 0 pairs, with ArchEmailsAllIssues and ArchIssuesAllEmails merged.

The amount of architectural labelled data also has influence on the similarity precision. We saw that the ArchEmailsAllIssues, with 620 architectural labelled emails, was outperformed by ArchIssuesAllEmails, with 1426 architectural labelled issues. This precision difference became even larger when we increase the architectural labelled issues to 2166 in iteration 4.

One notable occurrence that we found was that when the amount of ADDIE-pairs began to slow down, we did manage to find a lot of pair that were related on the same subject but not the same ADD. An example would be the issue HADOOP-1134[19] which says: "See recent improvement HADOOP-928 (that can add checksums to a given filesystem) regd more about it. Though this served us well there a few disadvantages" and ends by proposing a new ADD ("We propose to have CRCs maintained for all HDFS data in much the same way as in GFS."). If we look at the email with id 795[20] we see that they created the issue 'HADOOP-928'. The relation here is that the email with id 795 and issue 'HADOOP-928' are both about creating an ADD, whilst issue 'HADOOP-1134' then proposes a new ADD because of the other ADD. They are related on subject, but they are two different ADDs. We did not count this as an ADDIE-pair, but for future work this might be interresting to look into.

To summarize, our method that got us the best precision for ADDIE-pairs is to:

- Increase the architectural data set as much as possible.
- Remove all emails and issues with a word count smaller than 50.
- Calculate only similarity values for email issue pairs with a maximum creation time difference of 500.
- Calculate similarity values by taking the average of TF-IDF cosine similarity and SBERT embedding cosine similarity.

[19]issue url = https://issues.apache.org/jira/browse/HADOOP-1134
[20]email url = https://www.mail-archive.com/hadoop-dev@lucene.apache.org/msg07062.html

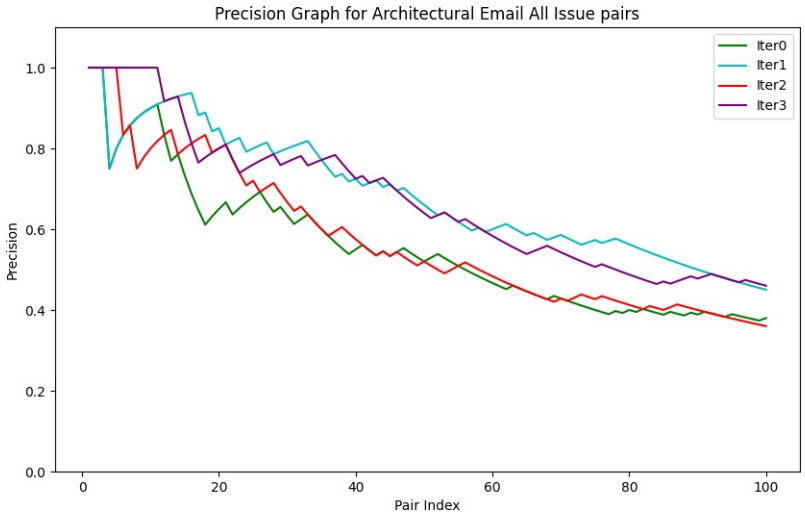

Figure 6: Precision graph of iterations 1 to 3 in Architectural Email All Issue pairs

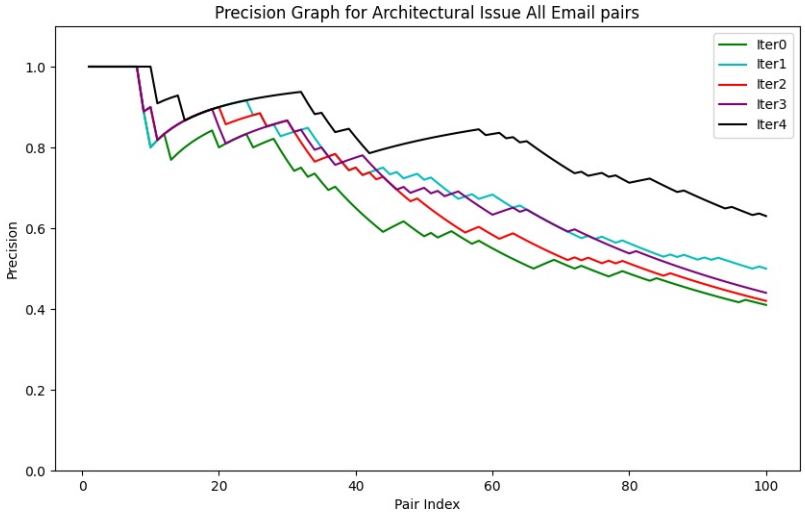

Figure 7: Precision graph of iterations 1 to 4 in Architectural Issue All Email pairs

4.3 RQ3: Characteristics of related issue - email thread pairs that discuss the same ADD

Now we have our 6 pattern, we can look if there are characteristics that set them apart from each other.

First we look at the amount of email per email thread. We know what patterns the ADDIE-pairs follow but we don't have a good estimate for how long certain discussion go on for. Looking at 8 we can see that pattern 5 has the most amount of emails per thread based on the IQR, but pattern 4 has the highest average. We can also see clearly that pattern 2 has a very small amount of emails per thread, which fits its pattern definition nicely. For pattern 4 we can justify the large amount of emails because it is about releasing and for every issue being released a discussion occur. Also there are a lot of replies which are simply about casting