Workflow IDseeqer

# Overview chart

Type of Data in the source database



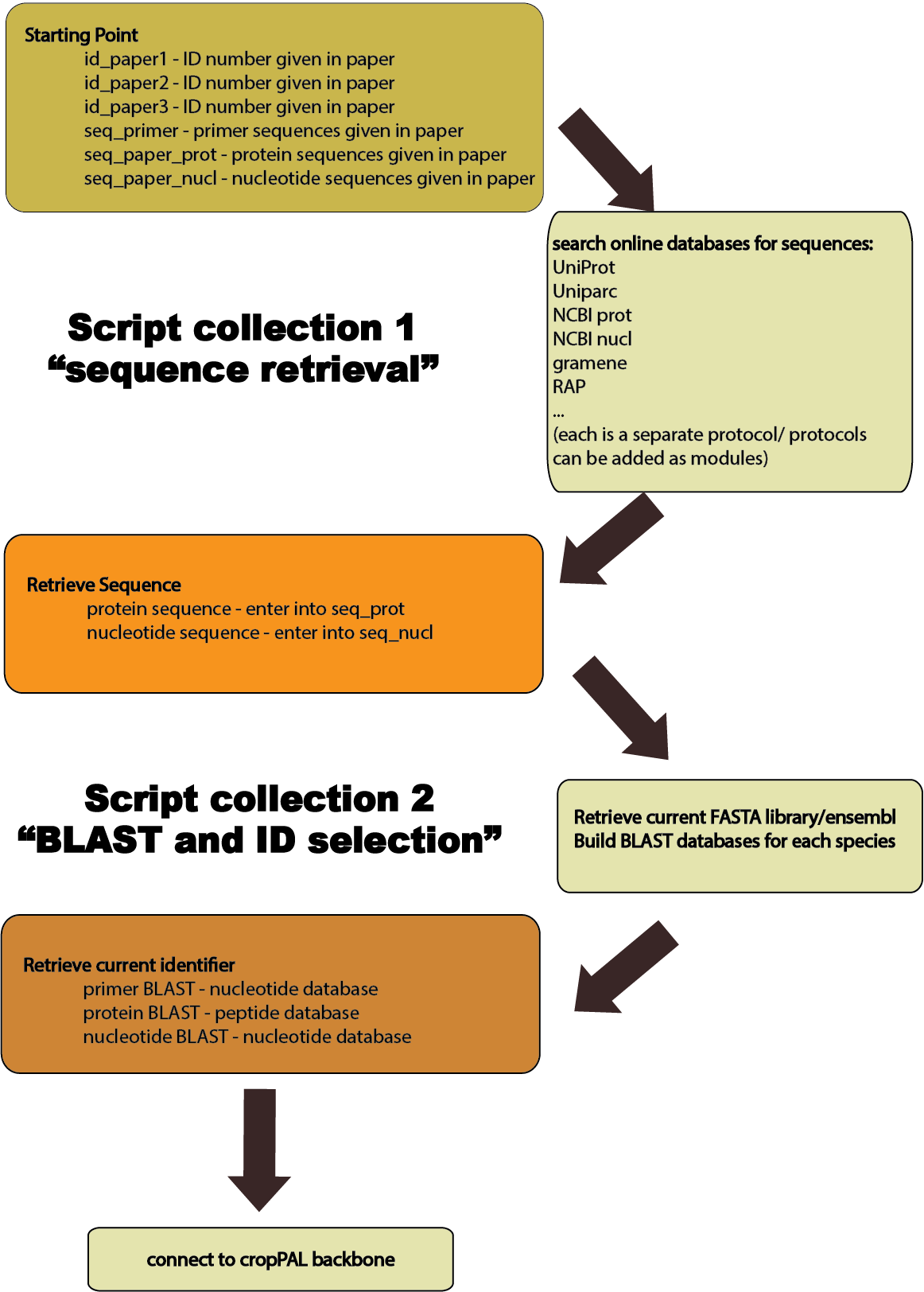
# Data filling

The curation data file must contain enough information to search for a sequence or provide a sequence in order to connect the data point.

If no sequence data is given the ids collected from the curation will be sent off to databases to retrieve sequences.

At this point several sequences can be retrieved. That means each data point can have a primer sequence, protein sequence, nucleotide sequence as well as ids. In the end the locus from difference sources can be compared if we want.

# Pipeline flowchart



# Set up description

The program will be an API available for scientists (or anyone) who requires a pipeline to retrieve a genetic sequence from a list of mixed IDs. Manually the user would have to take a single ID and submit it

This is an API that someone can download and then set up themselves. The user will install the BLAST and database packages before using the program. A template for collecting the sequence and loading in the IDs from a user table will be necessary. When starting the IDseeqer workflow the user will need a connection tap to enter the path to the database and password. This is only set up at the beginning of running the program and then the program will reconnect to the database automatically throughout the work flow. Since every update changes the table in the back, the program needs to update the memory.

The query: Since a user comes with a list of IDs there may not be a need for custom queries to the database. There can be a default query and an extra option for designing your own in case you know a lot of your entries in your database do not need to be sent anywhere for linking.

The strategy of sequence retrieval: A list of IDs exists with unknown content. The list of IDs could be batches of ID types. For example a list of 3000 IDs can be derived from 5 origins so that the first 2000 are Uniprot IDs and the rest is randomly mixed. Since this is often the case it would be good to have the following strategy: Based on the assumption that ID from the same origin are in concentrated batches the program should try to send the next ID to the same database where the last ID was successful. Thereafter ID1 will be sent to URL1 and if not successful to URL2 and so on. If update successful resume at the same place with ID2 and so on. If IDx is not successful start from the beginning of the URL hierarchy (URL1).

Retrieval Tracking: I would be useful to track 2 different instances (A) which URLs have been contacted already? (B) where was the retrieval successful? The update protocol will update the column "seq\_source" with a code or the origin description.

# Query and Collection parameters

ID lists: The IDs that are the queries derived from manual curation. There may be only one ID per entry but most commonly 2-4 IDs. Any of the IDs can be used and will be entered in order of suitability (manual estimation). These IDs for querying can be found in the columns id\_paper1 – id\_paper3.

Sequences: There is the thing with the difference between nucl (nucleotide) and prot (protein) sequences as well as where the sequence comes from. There may be one or multiple sequences to any entry in the database. Here the summary of the currently included cases:

Protein sequences (prot):

1. Protein sequence derived from the manual curation seq\_paper\_prot
2. Protein sequence derived from the sequence retrieval using the IDseeqer program seq\_prot

Nucleotide sequences:

1. Primer sequence derived from the manual curation seq\_primer
2. Nucleotide sequence derived from the manual curation seq\_paper\_nucl
3. Nucleotide sequence derived from the sequence retrieval ( IDseeqer program) seq\_nucl

There could be one or several of the sequences. Since we do not know beforehand what will be successful it is useful to fill up nucl and prot sequences. Since there is the high possibility of having a nuclear and a protein sequence it may be necessary to retain the columns for these.

Also they are different entities and need to be distinguishable. Afterwards it is really hard to distinguish them so the only way to do it is at the time of retrieval.

BLAST collection: The BLAST will yield a top scoring hit in the designed database of choice (version of Ensembl). Depending on the sequence BLASTed the retrieved locus will be a protein (prot) or cdna (nucl). A final protocol needs to be developed to have a coherent set of loci that are proteins. Currently there are the following collection columns:

1. Locus derived from the curation-derived sequences seq\_paper\_~ is collected in locus\_seq\_paper
2. Locus derived from the retrieved sequences (seq\_~) is collected in locus\_seq
3. Locus derived from primers (seq\_primer) is collected in locus\_primer

The collection allows to possible crosscheck the retrieved locus derived from several sources (as a confirmation) before proceeding with the final list. There are some species of plants that have distinct protein and transcript IDs. So the locus in the columns will be different to the proteome reference list. This will be translated using the species\_xref table. The latter is something that is made specifically for the cropPAL purpose and therefore the question is to include this at all. There are options to change this by:

1. Adding the xref backbone protocol to IDseeqer
2. Choosing the reverse BLAST protocol with 6-frame translation
3. Not worrying about it at all?

The above are all somewhaty painful.

The final step right now is to fill the locus column with protein IDs derived from the xref table.

# Sequence Retrieval Hierarchy

The program takes on entry (row ID) and collects id\_paper1

The id\_paper1 will be sent to the database in the following order:

1. Uniport – Retrieval: prot
2. NCBI prot – Retrieval: prot
3. Gramene – Retrieval: prot
4. UniParc – Retrieval: Uniparc ID => prot
5. RAP – Retrieval: gene (nucl)
6. NCBI nucl – Retrieval: nucl
7. NUCCORE – Retrieval: nucl

If at any of these stages a sequence is retrieved and the database is updated: Stop the search and send the next request.

# BLAST retrieval Hierarchy

Once all the sequences are retrieved there is another program that will start the ID retrieval (BLAST)

The program will obtain the sequence X and use the program Y in this order:

|  |  |  |  |
| --- | --- | --- | --- |
| order | Type of sequence (X) | Program (Y) | update column for final output |
| 1 | seq\_primer | Blast\_primer | Locus\_primer |
| 2 | Seq\_paper\_prot | BLAST\_prot | Locus\_seq\_paper |
| 3 | Seq\_paper\_nucl | BLAST\_nucl | Locus\_seq\_paper |
| 4 | Seq\_prot | BLAST\_prot | Locus\_seq |
| 5 | Seq\_nucl | BLAST\_nucl | Locus\_seq |