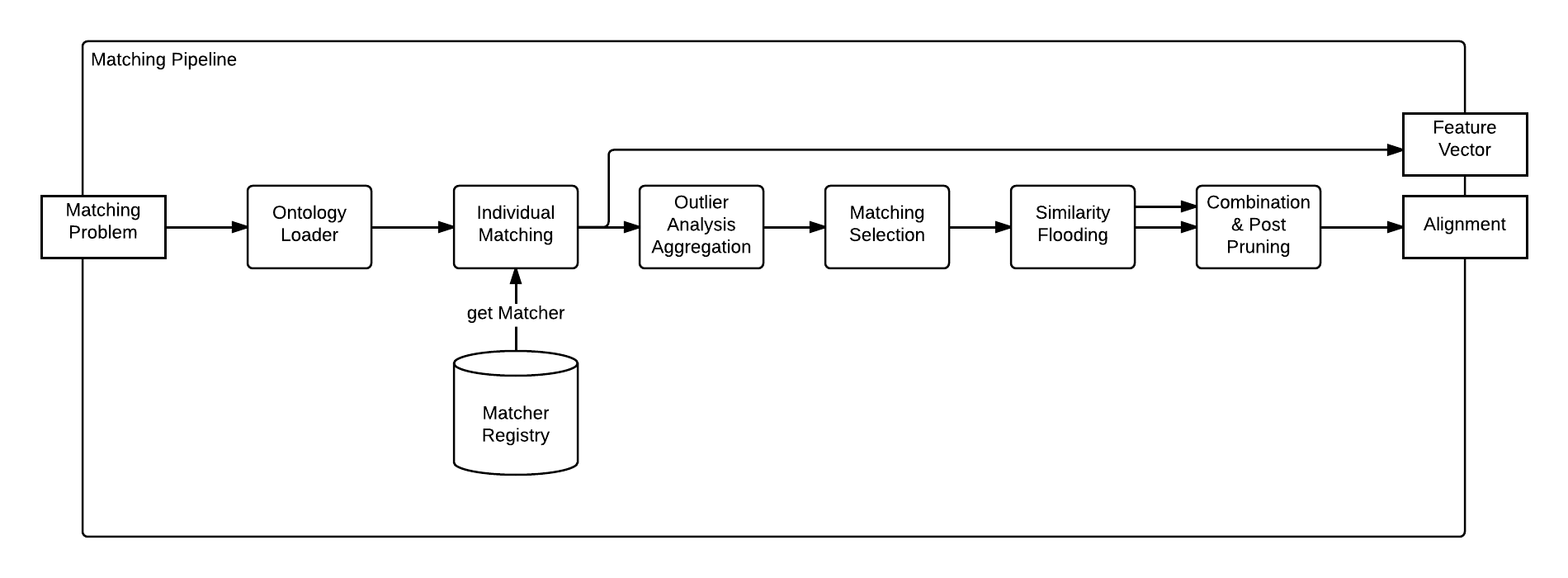
# Current State Master Thesis

## Concept for further development

Based on my literature review I redesigned the pipeline, to hopefully improve the results. Especially inspiration was taken from the AgreementMaker (light) System and Yam++, which were under the best performing systems over the last years in OAEI challenge.

Now the design of the current matching process is as follows: (not fully implemented yet)



### Individual Matching

Currently have a list of 21 base matchers, being in the category (string-based, language-based and structural matcher, comparing both local and global properties of an ontology). 19 are actually integrated.

## Aggregation

* The outlier analysis process is still the same as last time.
* Cluster Based Outlier Factor
* DBScan for clustering and afterwards factor computation
* TODO:
  + Change to X-Means
  + Separate aggregation for properties and classes

### Matching Selection

* Greedy Rank Selection(AML) and Greedy Selection(YAM++) used.
* Ensures 1 to 1 Cardinality of Matchings

### Structural and Individual Matching Combination

Combine them by only considering the matches from the similarity-flooding algorithm that were not discovered by the individual matchers.

### Post Pruning

* To a last verification step in order to exclude false matchings
* First implementation Step: Simple datatype comparision for datatype properties.
* Second implementation Step: Consider implementing a semantic verification step as in YAM++,
* Alternative implement a repair step, to remove incoherent alignments from the alignment (<http://arxiv.org/pdf/1307.5322v1.pdf> implemented in AML 2.0, or by Christian Meileckes Dissertation)

### Points for discussion:

* Similarity Propagation:
  + It could be treated differently. Instead of running it on the result of the aggregated individual matchers. It is possible to choose one arbitrary matcher and let the similarity propagation run on this results. A potential benefit for this could be that the result of the similarity propagation, could be included into outlier analysis detection
  + Another possible alternative is to perform outlier analysis aggregation twice, so once before similarity propagation and once more afterwards with similarity propagation
* Matching Efficiency.
  + The individual matching process can be easily parallelized
* Match datatype with object property allowed?
* Discard individual matching?