Progressive Entity Resolution: A Design Space Exploration

Jakub Maciejewski [1], Konstantinos Nikoletos [2], George Papadakis [1] and Yannis Velegrakis [2]

[1] Department of Informatics and Telecommunications, University of Athens, Greece
{sdi1700080,gpapadis}@di.uoa.gr
[2] Department of Information and Computing Sciences, Utrecht University, The Netherlands
{k.nikoletos,i.velegrakis}@uu.nl

Entity Resolution (ER) is typically implemented as a batch task that processes all available data before identifying duplicate records. However, applications with time or computational constraints, e.g., those running in the cloud, require a progressive approach that produces results in a pay-as-you-go fashion. Numerous algorithms have been proposed for Progressive ER in the literature. In this work, we propose a novel framework for Progressive Entity Matching that organizes relevant techniques into four consecutive steps: (i) filtering, which reduces the search space to the most likely candidate matches, (ii) weighting, which associates every pair of candidate matches with a similarity score, (iii) scheduling, which prioritizes the execution of the candidate matches so that the real duplicates precede the non-matching pairs, and (iv) matching, which applies a complex, matching function to the pairs in the order defined by the previous step. We associate each step with existing and novel techniques, illustrating that our framework overall generates a superset of the main existing works in the field. We select the most representative combinations resulting from our framework and fine-tune them over 10 established datasets for Record Linkage and 8 for Deduplication, with our results indicating that our taxonomy yields a wide range of high performing progressive techniques both in terms of effectiveness and time efficiency.

The basic idea of this work can be visualized with Figure 1. The horizontal axis corresponds to the verified pairs, while the vertical one corresponds to recall. We define the area under the curve as **progressive recall@N**, where N is the budget of the maximum verified pairs. It takes values in $[0, 1]$, with higher values indicating higher effectiveness. In other words, the higher the progressive recall is, the more and earlier are the existing duplicates detected.

In this context, our goal can be formally described as follows: Given two data sources, D_1 and D_2, along with a budget of N verifications, **Progressive Entity Matching** produces a set of candidate pairs ordered such that progressive recall@N is maximized, while the run-time is minimized.

Paper published and presented at ACM SIGMOD 2025 [1].

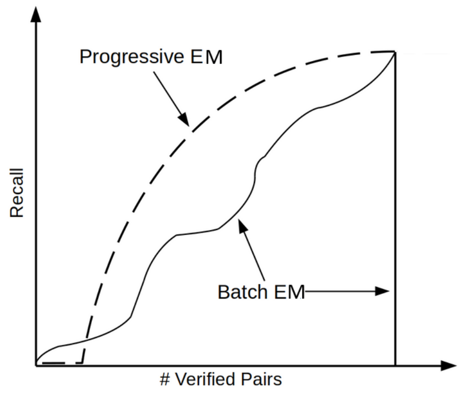

Figure 1: Batch vs Progressive Entity Matching.

References

[1] Maciejewski, J., Nikoletos, K., Papadakis, G., & Velegrakis, Y. (2025). Progressive Entity Matching: A Design Space Exploration. *Proceedings of the ACM on Management of Data (PACMMOD)*, 3(1), 65:1–65:25. `https://doi.org/10.1145/3709715`.