The results demonstrate that an efficient declarative schedule can be derived automatically for a difficult dependency problem.

The results also demonstrate, unintentionally, the benefit of QVTc' explicit trace model and the consequent linear performance when an appropriate cache is synthesized for the unnavigable opposite accesses. The quadratic performance of ATL, EMFTVM and QVTo highlights an implementation deficiency that can be remedied at the expense of some extra working memory. Ensuring that the extra memory cost is modest and the additional execution time is small, probably requires similar static compile-time or load-time analyses to those performed by Eclipse QVTd.

The code and results for this example are available in the tests/org.eclipse.qvtd.doc.exe2016.tests plugin of the https://git.eclipse.org/r/mmt/org.eclipse.qvtd GIT repository. Significant bugs in Eclipse QVTd were fixed to support this example. These fixes should be available in the Oxygen M2 milestone build in mid September 2016.

VIII. Related Work

Scheduling and particularly static scheduling has been a rich research topic with provision of optimized schedules recognized as a computationally hard problem. The many works of the Ptolemy group [1] that build upon [4] has been a strong background influence. However the appreciation that metamodels impose such strong constraints that sensible schedules can be produced rapidly for declarative transformations appears to be novel.

The Graph Transformation community has been very active in providing a rigorous foundation for graph mappings. Sadly the QVT specification ignored this important work, preferring instead to define the semantics of the QVTr transformation language using an incomplete exposition of a transformation of QVTr written in an untested QVTr to another language (QVTc) that has at best informal semantics. The utility and power of the QVTs graphical Micro-Mapping and its Model of Computation may begin to bridge the gap between these two communities. The automated coloring in QVTs is inspired by Henshin's [2] manual use of colors to denote create/delete/no-change in endogenous transformations. The reification of the QVTc traceability element mirrors the evolution operators in UMLX [7] for heterogeneous transformations.

Active Operations [3] also reify mappings to persist the state necessary for incremental execution. Micro-Mappings similarly support incremental execution, but their primary rationale is to be a deadlock-free unit of computation.

IX. Conclusion

We have introduced the Micro-Mapping Model of Computation and shown how it supports efficient declarative schedules for Eclipse QVTc and QVTr.

We used the Micro-Mapping Model of Computation to demonstrate the need for speculative creation of trace objects.

We have shown how a graphical presentation of metamodel and dependency analyses tames the naive inefficiencies of a declarative schedule.

We have introduced the first implementation of the QVTc specification.

We have presented the first results for a QVTr implementation using a direct code generator.

We have mentioned some future works. Many more optimizations to do.

Acknowledgment

The authors would like to thank Adolfo Sanchez-Barbudo Herrera, Horacio Hoyos Rodriguez, Dimitris Kolovos and Richard Paige for helpful discussions about declarative scheduling approaches. Horacio prepared some of the results and prototyped some of the scheduler algorithms.

References

[1] Bhattacharyya, S., Murthy, P., Lee, E.: Software synthesis from dataflow graphs, Kluwer Academic Press, Norwell, MA, 1996

[2] Biermann, E., Ermel, C., Schmidt, J., Warning, A.: Visual Modeling of Controlled EMF Model Transformation using HENSHIN Proceedings of the Fourth International Workshop on Graph-Based Tools, GraBaTs 2010.

[3] Jouault, F., Beaudoux, O.: On the Use of Active Operations for Incremental Bidirectional Evaluation of OCL 15th International Workshop on OCL and Textual Modeling, Ottawa, 2015

[4] Lee, E., Messerschmitt, D.: Synchronous data flow, Proceedings of the IEEE, 1987

[5] Lee, E., Sangiovanni-Vincentelli, A.: Comparing models of computation, Proceedings of the 1996 IEEE/ACM international conference on Computer-aided design

[6] OMG. Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification, Version 1.3. OMG Document Number: ptc/16-06-03, June 2016.

[7] Willink, E: UMLX : A Graphical Transformation Language for MDA Model Driven Architecture: Foundations and Applications, MDAFA 2003, Twente, June 2003. http://eclipse.org/gmt/umlx/doc/MDAFA2003-4/MDAFA2003-4.pdf

[8] Willink, E.: Yet Another Three QVT Languages. ICMT 2013 (2013)

[9] Eclipse ATL Project. https://projects.eclipse.org/projects/modeling.mmt.atl

[10] Eclipse EMF Project. https://projects.eclipse.org/projects/modeling.emf.emf

[11] Eclipse OCL Project. https://projects.eclipse.org/projects/modeling.mdt.ocl

[12] Eclipse QVT Declarative Project. https://projects.eclipse.org/projects/modeling.mmt.qvtd