The TTC 2014 Movie Database Case

Tassilo Horn
Institute for Software Technology
University Koblenz-Landau, Germany
horn@uni-koblenz.de

Christian Krause
SAP Innovation Center, Germany
christian.krause01@sap.com

Matthias Tichy
Chalmers | University of Gothenburg, Sweden
matthias.tichy@cse.gu.se

Social networks and other web 2.0 platforms use huge amounts of data to offer new services to customers. Often this data can be expressed as huge graphs and thus could be seen as a potential new application field for model transformations. However, this application area requires that model transformation tools scale to models with millions of objects. This transformation case targets this application area by using the IMDb movie database as a model. The transformation deals with identifying all actor couples which perform together in a set of at least three movies.

1 Introduction

The driving force behind social networks and other new web 2.0 offerings is often a huge amount of data from which interesting information can be extracted. Consequently, concepts like MapReduce [4] and libraries like Hadoop [2] and Giraph [1] have been developed to efficiently process this huge amount of data. However, model transformation approaches have not adressed this field so far.

Automotive software is an already well-established application field for model-driven software engineering and its models also approach huge sizes. As a consequence from these two examples, model transformation approaches must be scalable to models with million objects to be applicable for these application areas.

In the following, we present a case which uses the IMDb movie database [6] as a data source. The IMDb movie database contains information about movies, actors performing in the movies, movie ratings, etc. The main task is to develop a model transformation which identifies *all* couples of two actors who perform in at least three common movies and calculate the average rating of those movies.[1] This core task is then generalized to cliques of n actors. Furthermore, some queries calculating top-15 lists of couples and cliques are to be written. Evaluation criteria are correctness/completeness and performance.

In the next section, we describe the case in more detail including the meta model as well as the different core and extension tasks. After that, Section 3 presents the evaluation criteria for submitted solutions to this case.

2 Detailed Case Description

We use the IMDb data about movies, actors, actresses, and movie ratings for this transformation case. The resulting metamodel is shown in Figure 1. The names of actors and actresses are always unique in

[1] This task together with a solution in Henshin is also described in [7].

Submitted to:
TTC 2014

the models. In addition to the obvious classes, the metamodel contains a common superclass for actors and actresses as well as classes for groups of actors which play in common movies. The class Group contains the attribute avgRating which is intended to store the average rating of the common movies of the group of actors. The metamodel distinguishes between groups of two persons (a Couple) or a Clique of n persons to support the different tasks in this transformation case.

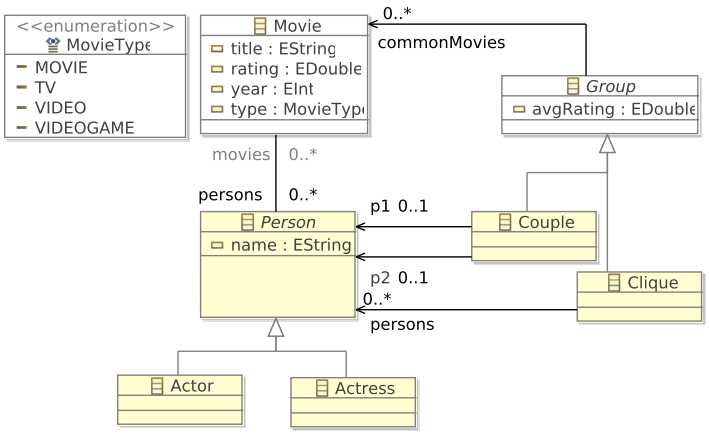

Figure 1: A metamodel for the relevant aspects of the movie database

The EMF model as well as a parser for the IMDb database files is available at [5]. The pre-parsed IMDb models are available on request[2]. The transformation case will use synthetic data (see Task 1) as well as real IMDb data for the evaluation of the correctness and performance of the submitted solutions.

2.1 Task 1: Generating Test Data

The goal of this task is to generate (synthetic) test data which will be used later to evaluate the correctness and the performance of the solution of the main task. The transformation to be implemented takes an empty model and a parameter $N \geq 0$, and generates movies, actresses, actors, and references between them. The number of objects to be generated is determined by the parameter N. Specifically, the transformation is supposed to generate $5N$ actors, $5N$ actresses and $10N$ movies, totalling in $20N$ nodes. Additionally, the references between persons and movies are generated in a specified way.

The specific patterns to be generated are shown in the Henshin [3] rules in Figure 2. Each of the two used rules generates five persons and five movies. The five persons in the rule createPositive play together in three movies. In contrast, every possible pair of persons generated by the rule createNegative plays together in at most two movies. The movie ratings and the name of the persons are derived from the rule parameter n which takes the values from 0 to $N-1$. The entry point for the test data generator is the iterated unit createExample. The unit has the integer-valued parameter N which determines the number of loops to execute. Specifically, this unit executes the sequential unit createTest N times with parameter values $0 \ldots N-1$ for n. The unit createTest invokes the rule createPositive and createNegative both with n as parameter. Test data should be generated for N being 1000, 2000, 3000, 4000, 5000, 10000, 50000, 100000, and 200000.

[2]Contact Matthias Tichy for getting access.

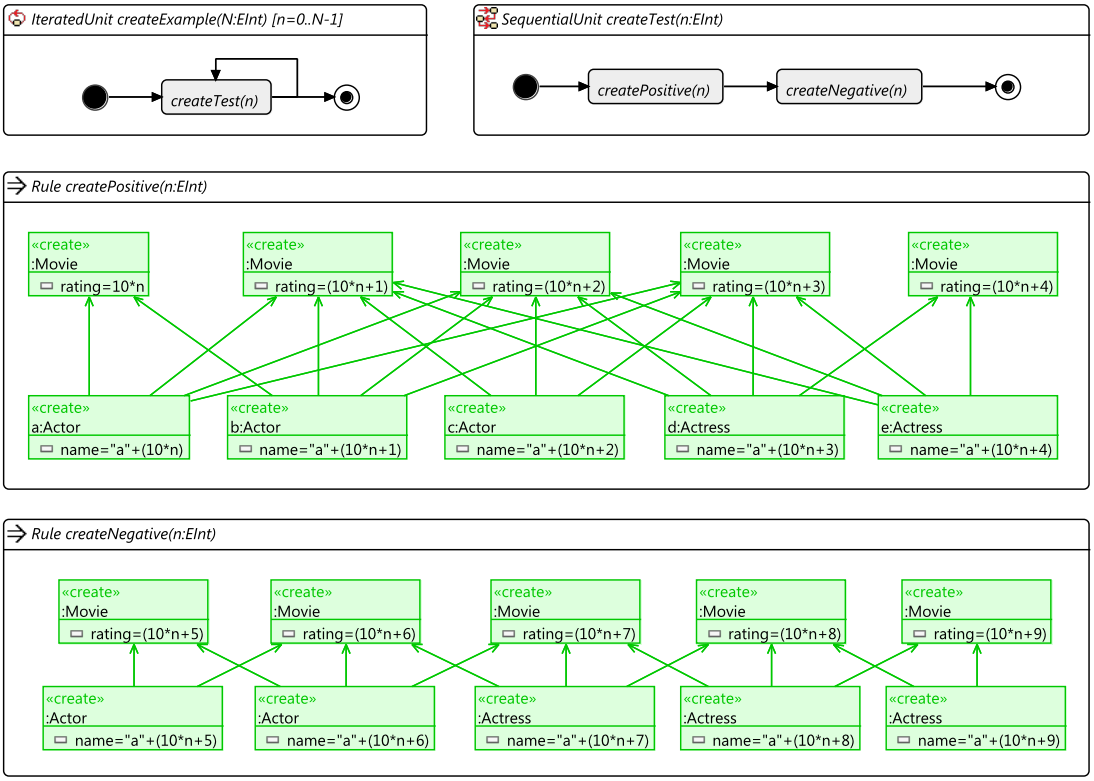

Figure 2: Henshin specification for generating synthetic movie test data.

2.2 Task 2: Finding Couples

In this task, a transformation shall be implemented that takes a graph consisting of inter-connected movies, actors and actresses as input, and creates additional nodes and links in this graph. Specifically, the task is to find all pairs of persons (actors or actresses) which played together in at least three movies. For every such pair, the transformation is supposed to create an object of type Couple referencing both persons using the p1 and p2 references, and referencing all movies in which *both* persons played in using the commonMovies reference.

2.3 Task 3: Computing Average Rankings

The input model of this task is the one generated in Task 2, i.e., a graph consisting of movie, actor, actress and couple nodes. The goal of this task is to set the avgRating-attribute of all couple nodes to the average (i.e. the arithmetic mean) of the ratings of all movies that *both* persons played in.

2.4 Extension Task 1: Compute Top-15 Couples

The goal of this task is to produce top-15 lists of the couples created by Task 2. For this purpose, two model queries should be given.

(a) Compute the top 15 couples according to the average rating of their common movies (requires Task 3 to be solved).

(b) Compute the top 15 couples according to the number of common movies.

Each of the couples in the top-15 lists should be printed with the names of the two persons, the average rating (only if Task 3 has been solved), and the number of the couple's common movies. We don't require printing the common movies' titles because the couple with the most common movies in the complete IMDb model has more than 400 of them. If two couples have the same value for the average rating/number of common movies, their order should be determined in some stable manner.

2.5 Extension Task 2: Finding Cliques

This extension task is a generalization of Task 2. A clique is a group of at least n persons (with $n \geq 3$) acting together in at least 3 movies. So a couple is essentially a clique of size $n = 2$.

The extension task is to find cliques of a given size n, and to create a Clique element for each of them referencing the clique's members using the persons reference and its common movies using the commonMovies reference.

The task will be evaluated for $n \in \{3,4,5\}$, so it could be solved by writing three similar rules manually. However, to achieve a full completeness score for this task, a solution should work for any $n \geq 3$. Therefore, a transformation could have n as a parameter, or there could be a higher-order transformation that receives n and generates a rule creating cliques of exactly that size.

2.6 Extension Task 3: Compute Average Rankings for Cliques

Like it was done for couples in Task 3, the avgRating attribute of cliques should be set to the average rating of all its common movies.

2.7 Extension Task 4: Compute Top-15 Cliques

This is a variant of Extension Task 1 for cliques instead of couples. Again, two queries should be given.

(a) Compute the top-15 cliques of a given size n according to the average rating of their common movies (requires Extension Task 3 to be solved).

(b) Compute the top-15 cliques of a given size n according to the number of common movies.

Again, every clique should be printed with the names of its members, the average rating, and the number of common movies.

3 Evaluation Criteria

The evaluation of the submitted transformation will be done on synthetic data as well as real data from the IMDb database. The IMDb database is regularly updated. In order to provide a common set of data, we provide the models generated from the IMDb database in December 2013 to participants by request[2].

3.1 Correctness and Completeness

All tasks and extension tasks are scored evenly with zero to three points. Zero means the task has not been tackled at all, three points means the task has been completely solved and the implementation is correct. The performance of a solution is not relevant, here. If a solution works correctly for the smaller models but won't terminate or run out of memory for the larger models, it may still achieve three points.

The following list explains the evaluation criteria for the individual tasks.

Task 1 The test data generation will be evaluated for different values of N. The correct number of elements, their relationships, and the correctly set attribute values will be assessed.

Task 2 The correct number of couples will be evaluated for both the synthetic and the IMDb models. Furthermore, the correct setting of the p1, p2, and commonMovies references will be spot-checked.

Task 3 The correct average ranking of couples will be checked for both synthetic and IMDb models.

Ext. Task 1 The Top-15 lists of couples will be evaluated for both the synthetic and the IMDb models.

Ext. Task 2 The finding of cliques will be evaluated for the sizes $n \in \{3,4,5\}$ for the synthetic models and for $n = 3$ for the IMDb models. The main criterion is that the value of n can be chosen freely, i.e., the rule for a given n is not written manually but it is a parameter to the transformation or a parameter to a higher-order transformation generating a transformation for that value.

Ext. Task 3 The correct average ranking of cliques will be checked for both synthetic and IMDb models.

Ext. Task 4 The Top-15 lists of cliques will be evaluated for both the synthetic ($n \in \{3,4,5\}$) and the IMDb models ($n = 3$). Like for Extension Task 1, the a stable sorting according to (a) average rating, or (b) number of common movies is enough to achieve a full score.

3.2 Benchmarks

The goal of this task is to generate a performance benchmark of your solution to Task 2 (finding couples). This benchmark should be executed using two different sets of input data:

(a) synthetic test data generated using the transformation for Task 1,

(b) provided data from the IMDb movie database (available at [6]; parsable, e.g., using [5]).

For both cases, you should run the transformation for Task 2 and measure the time needed to complete the transformation (without loading and saving the model). If you solved also the extension task 2 (finding cliques), please also generate benchmarks for these cases using $n \in \{3,4,5\}$ for the synthetic test models and $n = 3$ for the IMDb models.

In order to evaluate the scalability of the solution and to compare it with other solutions, you should use specific input models / model sizes. For the synthetic test data, please use the values for N stated in Section 2.1. For the IMDb data version, please use the provided models which are available on request[2].

The benchmark results are scored by comparing the run-times with the other solutions. The fastest solution gets 21 points (same amount as for correctness and completeness). The rest of the solutions get proportional scores.

References

[1] Apache Software Foundation. Apache Giraph. `http://giraph.apache.org`.

[2] Apache Software Foundation. Apache Hadoop. `http://hadoop.apache.org`.

[3] T. Arendt, E. Bierman, S. Jurack, C. Krause, and G. Taentzer. Henshin: Advanced concepts and tools for in-place EMF model transformations. In *Proc. MoDELS 2010*, LNCS 6394, pages 121–135. Springer, 2010.

[4] Jeffrey Dean and Sanjay Ghemawat. MapReduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, January 2008.

[5] Tassilo Horn. IMDB2EMF: `https://github.com/tsdh/imdb2emf`.

[6] Internet Movie Database (IMDB). Alternative interfaces: `http://www.imdb.com/interfaces`.

[7] Christian Krause, Matthias Tichy, and Holger Giese. Implementing graph transformations in the bulk synchronous parallel model. In *Proc. FASE 2014*. Springer, 2014. Accepted for publication.