Correctness of Incremental Model Synchronization with Triple Graph Grammars

Fernando Orejas* and Elvira Pino

Universitat Politècnica de Catalunya, Spain
{orejas,pino}@lsi.upc.edu

Abstract. In model-driven software development, we may have several models describing the same system or artifact, by providing different views on it. In this case, we say that these models are consistently integrated.

Triple Graph Grammars (TGGs), defined by Schürr, are a general and powerful tool to describe (bidirectional) model transformations. In this context, model synchronization is the operation that, given two consistent models and an update or modification of one of them, finds the corresponding update on the other model, so that consistency is restored. There are different approaches to describe this operation in terms of TGGs, but most of them have a computational cost that depends on the size of the given models. In general this may be very costly since these models may be quite large. To avoid this problem, Giese and Wagner have advocated for the need of incremental synchronization procedures, meaning that their cost should depend only on the size of the given update. In particular they proposed one such procedure. Unfortunately, the correctness of their approach is not studied and, anyhow, it could only be ensured under severe restrictions on the kind of TGGs considered.

In the work presented, we study the problem from a different point of view. First, we discuss what it means for a procedure to be incremental, defining a correctness notion that we call incremental consistency. Moreover, we present a general incremental synchronization procedure and we show its correctness, completeness and incrementality.

Keywords: Model Transformation, Model Synchronization, Triple Graph Grammars, Incremental Model Synchronization.

1 Introduction

In model-driven development, we may have several models describing the same system or artifact, by providing different views on it. Then, we say that these models are consistently integrated. Similarly, we say that two models are consistent if they are complementary descriptions of some system. In this context, given two integrated models, model synchronization is the problem of restoring consistency when one of these models has been updated by propagating that update to the other model. The same problem is also studied in other areas like databases or programming languages [11, 14, 9].

* This work has been partially supported by the CICYT project (ref. TIN2007-66523) and by the AGAUR grant to the research group ALBCOM (ref. 00516).
Triple Graph Grammars (TGGs) \cite{11,12} are a general and powerful tool to describe (bidirectional) model transformations. On the one hand, a TGG allows us to describe classes of consistently integrated models and, on the other hand, given some source model $M_1$, using the so-called derived operational rules associated to the TGG, we can find a corresponding consistent target model $M_2$. There are different approaches to describe model synchronization in terms of TGGs, but most of them have a computational cost that depends on the size of the given models. This may be rather inefficient since the given models may be large. To avoid this problem, Giese and Wagner \cite{4} have advocated for the need of incremental synchronization procedures, meaning that their cost should depend only on the size of the given update. In particular they proposed one such procedure. Unfortunately, the correctness of this approach is not studied and, anyhow, it could only be ensured under severe restrictions on the kind of TGGs considered, since the approach only works for the case when source and target models are bijective.

In this paper we address the problem from a different point of view. First, we discuss what it means for a procedure to be incremental. Specifically, given a derivation used to create a model and an update on it, we establish what does it means incrementality with respect to a consistent submodel not affected by the update. Essentially, it means that there exists a derivation that builds the new model preserving that consistent submodel. Then, this idea is formulated as a correctness notion, that we call incremental consistency. This may be considered a first contribution of the paper.

Our second and main contribution is the introduction of a new general incremental synchronization procedure. In principle, the input for this procedure would be given by an integrated model $G$, a derivation of $G$ representing its structure, and an update on the source model of $G$. However, since storing a derivation may be expensive in terms of the amount of storage needed, we replace the derivation by dependence relations on the elements of $G$ that are shown to be equivalent, in an adequate sense, to the derivation. Specifically, we prove a theorem (Th.1 in Sec.4) that guarantees that the largest consistent submodel not affected by the update can be obtained from that dependencies without cost depending on the model. Then, the procedure consists of five steps. In the first one, based on the above result, we identify the part of the model that needs to be reconstructed and we mark all the elements that may need to be deleted. In the second step, if needed, we enlarge the part of the model that needs to be reconstructed. As we will discuss, this second step is only needed in some cases when the update does not allow incremental consistency with respect to the largest consistent submodel not affected by the update, but with respect to a smaller one. In the third step, following the same idea presented informally in \cite{5}, we build a model that is already consistent, by applying a variation of forward translation rules \cite{8,6} allowing us to reuse most relevant information from the target model. For this reason, we call these rules forward translation rules with reuse. However, the resulting model may not include elements from the target model that do not have a correspondence in the source model. To avoid this, in the fourth step we recover these elements by just using our dependence relations. Finally, in the fifth step we effectively delete target elements that are still marked to be deleted. We prove that the results of this procedure are always incrementally correct and complete in the sense that, if there is an incrementally correct solution, the procedure will find it.