Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 1995 Proceedings

Americas Conference on Information Systems (AMCIS)

8-25-1995

Formal Validation of Schema Clustering for Large Information Systems

Robert Winter
Institut für Wirtschaftsinformatik, Johann Wolfgang Goethe -Universität, winter@wiwi.uni-frankfurt.d400.de

Formal Validation of Schema Clustering for Large Information Systems

Robert Winter
Institut für Wirtschaftsinformatik, Johann Wolfgang Goethe - Universität
Box 11 19 32, 60054 Frankfurt am Main, Germany
winter@wiwi.uni-frankfurt.d400.de

1 Introduction

To become usable for documentation and visualisation purposes, the conceptual schema of a large information system (IS) has to be clustered. [1] But most of the "clustering", "abstraction", or "concentration" rules proposed in [1]-[6] are substantiated only intuitively and have not been validated formally. As a consequence, the application of these rules to large IS schemata leads to inconsistencies (e.g. cyclic references), unnecessary loss of information (e.g. arbitrary object type subsets), and/or impractical results (e.g. combinatorical explosion). In this paper, the NF2 relational model [7] is used to validate the application of clustering operations to conceptual schemata. Based on this validation concept and a critical review of schema clustering literature, some validated, general clustering rules are presented. The schema clustering concept is extended to the clustering of textual documentations. To prove the concept's feasibility, conceptual schemata and textual documentations of SAP's R/3 modules "Production Planning" and "Sales & Distribution" have been clustered.

2 Validation of Schema Manipulations

The Structured Entity Relationship (SER) model [8] provides a framework for the graphical arrangement and the formal validation of conceptual schemata for large ISs. By replacing relationship types with dependent object types and references, additional semantics are introduced into conceptual modeling, and the resulting schemata can be processed as directed graphs. As an example, the "structure" provided by directed references can be used for the graphical arrangement of schema elements. Since dependent object types can always be found right of or below the referenced object type(s), dependencies within large conceptual schemata can be visualised, and the represented semantics are accessible in a specific, convenient form.

For every SER schema, an equivalent set of NF2 relation types must exist. Therefore, conceptual modeling can at least partially be validated by a formal proof of certain properties that these NF2 relation types must have. [8] Due to its obvious advantages with regard to visualisation and validation, a modified SER model is used by SAP to develop and maintain the R/3 system. In contrast to refinement operations during schema development, however, the SER model does not support post-development clustering operations.

Our approach to IS schema clustering is based upon the correspondence of SER schemata with NF2 schemata. Formal modeling cannot and should not replace conceptual modeling. But by formal validation, the consistency of modeling operations can be guaranteed.

Since an equivalent set of NF2 relation types must exist for every SER schema, it must also exist for clustered SER schemata. A schema manipulation that transforms the detailed SER schema Sd to a clustered SER schema Sc, therefore, implies a schema manipulation that transforms a set of normalised relation types S'd into a set of NF2 relation types S'c. In contrast to the manipulation of the conceptual schema, the manipulation of the relational schema can be validated formally using NF2 calculus. Hence, clustering of SER schemata must follow certain rules that result from NF2 algebra. [9] The clustering validation concept is illustrated by figure 1.

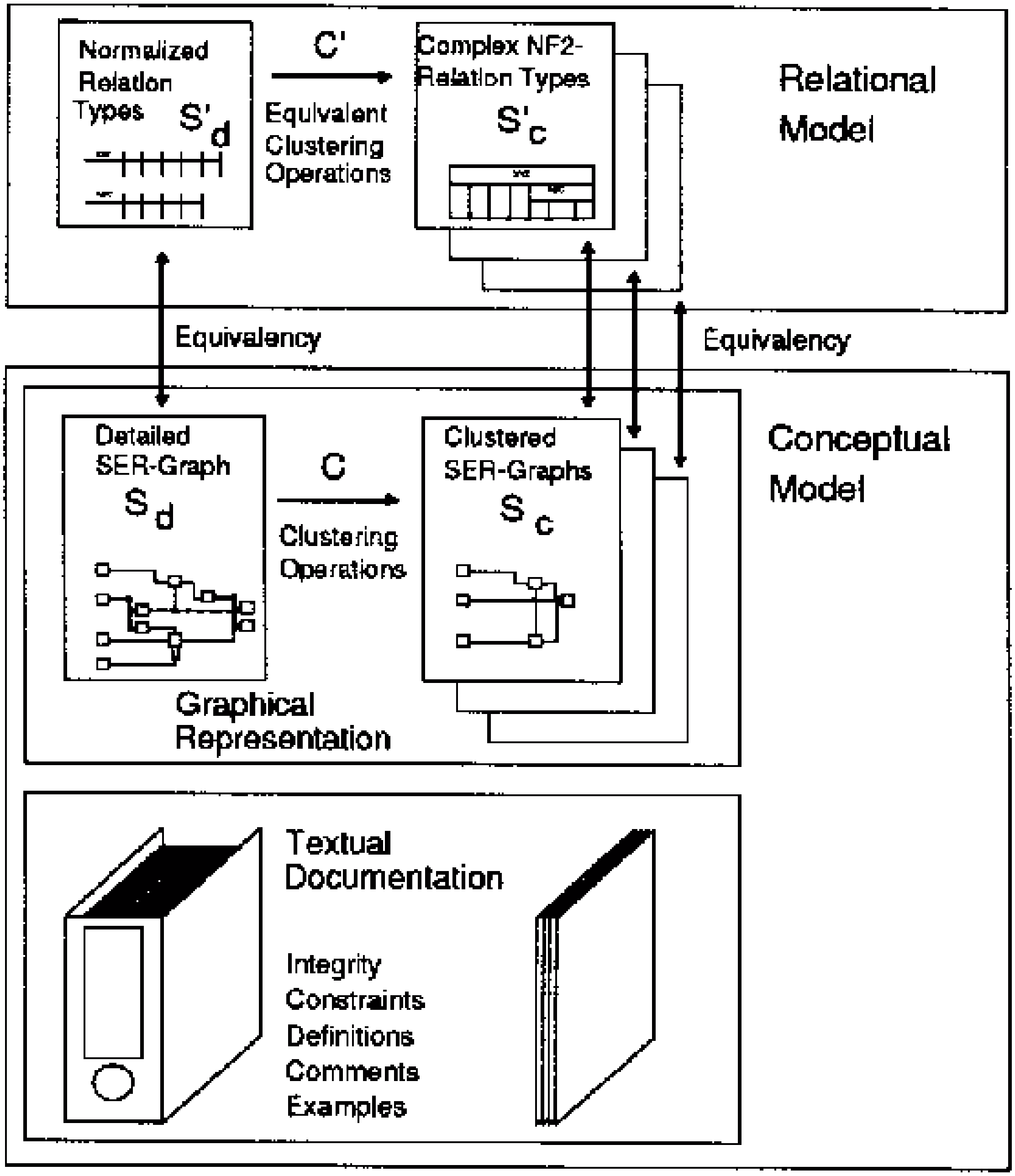

Figure 1. Clustering operations in conceptual model and in relational model [9]

A SER schema clustering operation is considered valid if and only if a sequence of join and nesting operations can be identified by which the NF2 schema that corresponds to the detailed SER schema is transformed into a NF2 schema that corresponds to the clustered SER schema. For SER schema clustering, however, the formal rules imposed by NF2 calculus have to be translated into conceptual modeling rules.

3 Conceptual Clustering Rules

In this section, the most important clustering rules proposed in [1]-[6] are reviewed. Applying our validation concept, many of these rules have to be rejected, and only few can be accepted unchanged or with minor modifications. The discussion yields a set of formally validated clustering principles.

Rule 1: **Aggregation and selection may both be used for schema clustering.**

While in most studies only aggregation is used to derive object types of the clustered schema [1][4]-[6], in [2] only selection of "key concepts" is allowed. If object type clustering is limited to aggregation, subsequent reference clustering may create cyclic references without semantical meaning. In NF2 calculus, aggregation as well as selection can be formalised by join and nesting operations. [9] Therefore, only the combined aggregation/selection approach proposed in [3] can be validated.

Rule 2: **Aggregation must not create new object types.**

While in [3] always new names are assigned to object types that result from an aggregation, other studies restrict the assignment of new names to certain conditions [1][4]. Only in [5] and [6], aggregate object types must be named after one of their components. By NF2 nesting operations, new NF2 relation types cannot be created. Therefore, the naming rule in [5][6] can be validated, and all other naming rules have to be rejected. As a consequence, the detailed conceptual schema must comprise the entire set of aggregate object types that result from subsequent clustering operations.

Rule 3: **Aggregation must be based on references.**

Most studies require aggregations to be based on references. [1][3][5][6] However, in [4] an example can be found where object types are aggregated that are neither directly nor indirectly connected by references. This operation must be considered as invalid because in the corresponding NF2 schema, no transformation can be defined that yields a relation type that corresponds to the aggregate object type. From a conceptual point of view, the aggregate object type in [4] combines incompatible information and, therefore, has no semantic meaning.

Rule 4: **Object type clustering cannot be performed algorithmically.**

Many studies propose a clustering procedure that is at least partially automatised: "Key concepts" are determined using a graphical analysis [2], "logical horizons" are automatically identified to select "major entity types" [3], and even a completely automatical clustering algorithm is proposed [5]. While the remaining studies utilise at least some heuristics to support clustering decisions to some extent [1][4], only in [6] the primacy of semantic considerations for the clustering of conceptual schemata is emphasised. Algorithms and/or heuristics can only consider syntactic properties like the cardinality of references [5] or the conherence of the schema graph [2]. But "dominant", "major", or "key" object types cannot be identified based only on syntactical properties: Often syntactically "weak" object types turn out to be semantically dominant, or object types that connect "key concepts" turn out to be semantically weak. Therefore, the utilisation of algorithms is limited to the validation of clustering decisions that have been proposed by a developer.

Rule 5: **An object type may be aggregated into more than one cluster.**

One the one hand, most studies do not allow to aggregate a semantically weak object type into more than one dominant object type. [1][3][4][6] But it is generally accepted that conceptual schemata are nets and not trees. If references to several, equally dominant object types exist, a general ban on multiple aggregation would cause the clustered schema to represent only an arbitrary subset of important semantical information. Therefore, it should be possible to aggregate an object type into all referenced (or referencing) dominant types.

On the other hand, the clustering algorithm proposed in [5] forces (syntactically) weak object types to be aggregated into all referenced object types. Because no selection of dominant types is made, the clustering algorithm causes the number of references to explode, and the clustered schema is loaded with unimportant information.

From a formal point of view, a relation type may be nested into more than one other relation types. Hence, it must be possible to aggregate semantically weak object types into all appropriate dominant object types. [9] The identification of dominant types again is a semantic decision and cannot be performed automatically.

Rule 6: **Reference clustering depends on object type clustering.**

Since references connect object types, object type clustering requires subsequent reference clustering. In case of object type aggregation, references of semantically weak object types are "inherited" by dominant types. [3][5] In case of object type selection, ingoing and outgoing references of ignored object types have to be concatenated. [6] References between object types within the same aggregate, and references between ignored object types are also ignored. [6] With a decreasing number of object types, however, the clustering process creates a lot of parallel references between remaining object types. In addition to "inheritance" and concatenation, therefore, it is necessary to aggregate parallel references between object types into one single reference and/or to ignore weak references. [9]

While the least restrictive reference prevails when references are concatenated, the most restrictive reference prevails when parallel references are aggregated. Since reference semantics are sufficiently represented by cardinality and participation constraints, it is possible to utilise an automatical procedure for reference concatenation and reference aggregation. Cardinality tables for reference aggregation can be found in [5] and [6]. These tables have been complemented in [9] to cover special reference types used in SAP's SER model extension.

4 Clustering of Textual Documentation

The creation of textual documentations for conceptual schemata is usually much more time-consuming than the creation of the schema itself. For a SER schema, a textual documentation includes at least definitions for all object types and reference types. In industrial applications, it is very important whether or not an individual textual documentation has to be provided for every clustered schema. Since a clustered schema is only a medium to facilitate the interpretation of its detailed counterpart, the only decisive description of the real world problem is the detailed conceptual schema and its detailed textual documentation [3]. Therefore, clustered schemata and their textual documentation should only be used as a supplement to their detailed counterparts.

Since every object type of a clustered schema must also belong to the detailed schema (rule 2), definitions for detailed object types can be reused in textual documentations of clustered schemata. However, it might be necessary to complement definitions of "surviving" major object types by definitions of those minor object types that have been incorporated by object type aggregation. This "definition assembly" can be avoided when for every object type, not only its definition, but also its references to other object types are included in the textual documentation of the detailed schema. In this case, the definition of clustered object types can be assembled easily by the user, and no redundant definitions have to be created and held consistent for the clustered schema.

Unfortunately, creating definitions for clustered references is not that simple. When a reference to a minor object type has been inherited by a major object type, it cannot necessarily be interpreted semantically without knowing the definition of the minor object type (that is no element of the clustered schema). For every clustered reference type, therefore, it is necessary to document which minor type has inherited the respective reference to the clustered object type. While it is easy to locate information on components of a clustered object type in the textual documentation of the detailed schema, it is difficult or even impossible to find out which components have been aggregated into a clustered reference type. For that reason, textual documentations of clustered schemata have to comprise "local" definitions for every clustered reference type. In addition to the definition of the reference type's semantic meaning, the corresponding elements of the detailed schema have to be mentioned.

5 Application to SAP's R/3 System

The clustering concept proposed in this paper has been developed for clustering the PP- (Production Planning) and the SD- (Sales & Distribution) SER subschema of SAP's R/3 system. For every subschema, the first of three planned clustering levels was generated. By clustering the PP subschema, the number of object types was reduced from 325 to 94 and the number of reference types was reduced from about 650 to about 190. By clustering the SD subschema, the number of object types was reduced from 365 to 141 and the number of reference types was reduced from about 850 to about 280.

The schema clustering turned out to be very time-consuming. But in contrast to semantic analysis and object type clustering which could be conducted very easily due to an excellent documentation, manual chaining and aggregation/selection of references became the main clustering effort. It would be possible to reduce this effort significantly by an automatic reference clustering algorithm: The directed graph structure of SER schemata can conceptually be interpreted as a multi-level bill of material. Like secondary demands can be derived by using a transitive closure of the multi-level bill of material, chained references can be derived by using a transitive closure of the SER schema. The resulting set of chained references represents the enumeration of all join and nesting operations on the respective NF2 relation types. A formal description of this procedure and its application to generate propagation paths for data manipulations is described in [10].

References

[1] Teorey, T.J. / G. Wei / D.L. Bolton / J.A. Koenig: ER Model Clustering as an Aid for User Communication and Documentation in Database Design, Communications of the ACM, 32 (1989), 8, 975-987

[2] Vermeir, D.: Semantic Hierarchies and Abstractions in Conceptual Schemata, Information Systems, 8/2 (1983), 117-124

[3] Feldman, P. / D. Miller: Entity Model Clustering: Structuring a Data Model by Abstraction. The Computer Journal, 29 (1986), 4, 348-360

[4] Jaeschke, P. / A. Oberweis / W. Stucky: Extending ER Model Clustering by Relationship Clustering, in: Elmasri, R. et al. (Eds): Entity Relationship Approach - ER'93, Berlin etc.: Springer 1994, 451-462

[5] Rauh, O. / E. Stickel: Entity Tree Clustering - A Method for Simplifying ER Design, in: Pernul, G. / A.M. Tjoa (Eds.): Entity Relationship Approach - ER'92, Berlin etc.: Springer 1992, 62-78

[6] Mistelbauer, H.: Concentration of Data Models - From Project Data Models to an Enterprise-Wide Data Architecture, Wirtschaftsinformatik, 33 (1991), 4, 289-299 (In German)

[7] Schek, H.-J. / M. Scholl: The Relational Model with RelationValued Attributes, Information Systems, 11 (1986), 2, 137-147

[8] Sinz, E.J.: The Structured Entity-Relationship Model (SERM), Angewandte Informatik, 30 (1988), 5, 191-202 (In German)

[9] Boßhammer, M. / R. Winter: Formal Validation of ER Model Clustering Operations, Research Report 94-10, Institut für Wirtschaftsinformatik, Johann Wolfgang Goethe Universität Frankfurt am Main

[10] Winter, R.: Formalised Conceptual Models as a Foundation of Information Systems Engineering, in: Loucopoulos, P. (Ed.): Entity-Relationship Approach - ER'94, Berlin etc.: Springer 1994, 437-455