Spectre

Wolfgang May
Institut für Informatik,
Universität Freiburg
Am Flughafen 17
D-79110 Freiburg
Federal Republic of Germany
may@informatik.uni-freiburg.de

2000/11/09

Abstract

The Spectre style allows running colors or fading out and fading in in a text. In addition to change the color with every letter, it is possible to change it with every word, every *group*, every line, or with every line in a tabular. The style requires the dvips package.

1 The User-Interface

The following table is typeset by ... [1] In Germany, the last three teams are relegated. Kaiserslautern managed to be re-promoted the subsequent year and become champion two years later.

[1] InGermany,thelastthreeteamsarerelegated.Kaiserslauternmanagedtobere-promotedthesubsequentyearandbecomechampiontwoyearslater.

1.	BV 09 Borussia Dortmund	34	19	11	4	76-38	+38	68
2.	FC Bayern München	34	19	5	10	66-46	+20	62
3.	FC Schalke 04	34	14	14	6	45-36	+ 9	56
4.	Borussia Mönchengladbach	34	15	8	11	52-51	+ 1	53
5.	Hamburger SV	34	12	14	8	52-47	+ 5	50
6.	FC Hansa Rostock	34	13	10	11	47-43	+ 4	49
7.	Karlsruher SC	34	12	12	10	53-47	+ 6	48
8.	TSV München 1860	34	11	12	11	52-46	+ 6	45
9.	SV Werder Bremen	34	10	14	10	39-42	- 3	44
10.	VfB Stuttgart	34	10	13	11	59-62	- 3	43
11.	SC Freiburg	34	11	9	14	30-41	-11	42
12.	1. FC Köln	34	9	13	12	33-35	- 2	40
13.	TSV Fortuna Düsseldorf	34	8	16	10	40-47	- 7	40
14.	TSV Bayer 04 Leverkusen	34	8	14	12	37-38	- 1	38
15.	FC St. Pauli Hamburg	34	9	11	14	43-51	- 8	38
16.	1. FC Kaiserslautern	34	6	18	10	31-37	- 6	36
17.	Eintracht Frankfurt	34	7	11	16	43-68	-25	32
18.	Krefelder FC Uerdingen	34	5	11	18	33-56	-23	26

WORDS:

eins zwei drei vier fuenf

T r a c i n g this concept one step further, also the initial states of procedures are not total interpretations, but their extension is inherited from the state-as-is interpretation of the calling state.

T r a c i n g this concept one step further, also the initial states of procedures are not total interpretations, but their extension is inherited from the state-as-is interpretation of the calling state.

EXPRESSIONS:

alpha beta gamma delta e f g h

alpha beta gamma delta e f g h

alpha beta gamma delta e f g h

LETTERS:

T r a c i n g this concept one step further, also the initial states of procedures are not total interpretations, but their extension is inherited from the state-as-is interpretation of the calling state.

T r a c i n g this concept one step further, also the initial states of procedures are not total interpretations, but their extension is inherited from the state-as-is interpretation of the calling state.

T r a c i n g this concept one step further, also the initial states of procedures are not total interpretations, but their extension is inherited from the state-as-is interpretation of the calling state.

T r a c i n g this concept one step further, also the initial states of procedures are not total interpretations, but their extension is inherited from the state-as-is interpretation of the calling state.

00T r a c i n g this concept one step further, also the initial states of procedures are not total interpretations, but their extension is inherited from the state-as-is interpretation of the calling state.

T r a c i n g this concept one step further, also the initial states of procedures are not total interpretations, but their extension is inherited from the state-as-is interpretation of the calling state.

Referential actions are specialized triggers used to automatically maintain referential integrity. While their local behavior can be grasped easily, it is far from clear what the combined effect of a set of referential actions, i.e., their global semantics should be. For example, different execution orders may lead to ambiguities in determining the final set of updates to be applied. To resolve these problems, we propose an abstract logical framework for rule-based maintenance of referential integrity: First, we identify desirable abstract properties like *admissibility* of updates which lead to a non-constructive global semantics of referential actions. We obtain a constructive definition by formalizing a set of referential actions RA as logical rules, and show that the declarative semantics of the resulting logic program P_{RA} captures the intended abstract semantics: The well-founded model of P_{RA} yields a unique set of updates, which is a safe, sceptical approximation of the set of all maximal admissible updates; the third truth-value *undefined* is assigned to all controversial updates. Finally, we show how to obtain a characterization of all maximal admissible subsets of a given set of updates using certain maximal stable models. Finally, we show how to obtain a characterization of all maximal admissible subsets of a given set of.

We study the following problem: Given a relational database D, a set of user-defined update requests , and a set of referential actions RA, find those sets of updates Δ which (i) preserve referential integrity in the new database D', (ii) are maximal wrt. and (iii) reflect the intended meaning of RA.

The problem is important both from a practical and theoretical point of view: Referential integrity constraints are a central concept of the relational database model and frequently used in real world applications. *Referential actions* (rac's) are specialized triggers used to automatically maintain referential integrity [?]. While the *local* behavior of rac's is quite intuitive and easy to understand, it is far from clear what their *global semantics* should be. In particular, different execution orders of rac's may lead to different outcomes, i.e. to *ambiguities* in determining the above Δ and D'.

Due to their practical importance, rac's have been included in the SQL2 standard and SQL3 proposal [?, ?]. However, the standards describe the meaning of rac's in a lengthy and procedural way, making it difficult to understand or predict their global behavior. The problem of ambiguous global semantics is "solved" by fixing a rather ad-hoc run-time execution model [?, ?]. In a different approach, [?] presents safeness conditions which aim at avoiding ambiguities at the schema level. However, as shown in [?], it is in general undecidable whether a database schema with rac's is ambiguous. Summarizing, the problem is complex and, from a theoretical point of view, has not been solved in a satisfactory way. We study the following problem: Given a relational database D, a set of user-defined update requests , and a set of referential actions RA, find those sets of updates Δ which (i) preserve referential integrity in the new database D', (ii) are maximal wrt. and (iii) reflect the intended meaning of RA.

The problem is important both from a practical and theoretical point of view: Referential integrity constraints are a central concept of the relational database model and frequently used in real world applications. *Referential actions* (rac's) are specialized triggers used to automatically maintain referential integrity [**?**]. While the *local* behavior of rac's is quite intuitive and easy to understand, it is far from clear what their *global semantics* should be. In particular, different execution orders of rac's may lead to different outcomes, i.e. to *ambiguities* in determining the above Δ and D'.

Due to their practical importance, rac's have been included in the SQL2 standard and SQL3 proposal [**?**, **?**]. However, the standards describe the meaning of rac's in a lengthy and procedural way, making it difficult to understand or predict their global behavior. The problem of ambiguous global semantics is "solved" by fixing a rather ad-hoc run-time execution model [**?**, **?**]. In a different approach, [**?**] presents safeness conditions which aim at avoiding ambiguities at the schema level. However, as shown in [**?**], it is in general undecidable whether a database schema with rac's is ambiguous. Summarizing, the problem is complex and, from a theoretical point of view, has not been solved in a satisfactory way.

In contrast to previous work, we present an abstract logical framework for rule-based maintenance of referential integrity: After introducing a generic language for rac's, we identify general abstract properties which a set of updates Δ wrt. a given set of rac's RA may possess (Section **??**). These abstract properties give rise to a natural but non-constructive global semantics. To obtain a constructive definition, we associate with every set of rac's RA a *logic program* P_{RA} (Section **??**), and show that the declarative semantics of P_{RA} captures the abstract semantics (Section **??**). This solves the above-mentioned problem in a rigorous and comprehensive way.

In contrast to previous work, we present an abstract logical framework for rule-based maintenance of referential integrity: After introducing a generic language for rac's, we identify general abstract properties which a set of updates Δ wrt. a given set of rac's RA may possess (Section **??**). These abstract properties give rise to a natural but non-constructive global semantics. To obtain a constructive definition, we associate with every set of rac's RA a *logic program* P_{RA} (Section **??**), and show that the declarative semantics of P_{RA} captures the abstract semantics (Section **??**). This solves the above-mentioned problem in a rigorous and comprehensive way.

In contrast to previous work, we present an abstract logical framework for rule-based maintenance of referential integrity: After introducing a generic language for rac's, we identify general abstract properties which a set of updates Δ wrt. a given set of rac's RA may possess (Section **??**). These abstract properties give rise to a natural but non-constructive global semantics. To obtain a constructive definition, we associate with every set of rac's RA a *logic program* P_{RA} (Section **??**), and show that the

declarative semantics of P_{RA} captures the abstract semantics (Section **??**). This solves the above-mentioned problem in a rigorous and comprehensive way.

Spektrum von Datenbank-Regelsprachen