Automatically Inferring ClassSheet Models from Spreadsheets

Jácome Cunha João Saraiva Martin Erwig

Universidade do Minho Oregon State University

VL-HCC 2010 - Madrid

An Example

◇	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	com_code	upc	description	size	case	nitem	store	week	move	qty	price	onsale	profit	ok
2	653	1111140009	DOVE DISH LIQUID	42 OZ	9	2851281	100	383	16	1	2.19		7.01	1
3	653	1111140009	DOVE DISH LIQUID	42 OZ	9	2851281	100	384	7	1	2.19		3.07	1
4	653	1111140009	DOVE DISH LIQUID	42 OZ	9	2851281	100	385	15	1	2.19		6.57	1
5	...	...	...	...	...	...	...	...	...	...	...	...	...	...
6	654	1111165003	SUNLIGHT AUTO GEL	88 OZ	6	2857061	100	390	6	1	3.75		4.50	1
7	654	1111165003	SUNLIGHT AUTO GEL	88 OZ	6	2857061	100	391	11	1	3.39	S	7.46	1
8	654	1111165003	SUNLIGHT AUTO GEL	88 OZ	6	2857061	100	392	8	1	3.75		6.00	1
9	...	...	...	...	...	...	...	...	...	...	...	...	...	...

- This spreadsheet (SS) contains information about detergents
- Cells contain values and formulas

Models...

- Model-driven engineering uses models to help programmers
- But in spreadsheets models are usually missing (e.g. legacy spreadsheets)
- It is difficult for end users to define the business model of a spreadsheet
- Inferring them from data is a possible solution

ClassSheet Models

- Object-oriented like specifications
- Contain spacial information, crucial for spreadsheets
- They are appropriate to specify the business logic of a spreadsheet

	A
1	**Income**
2	**Item**
3	value = 0
⋮	
4	Total
5	total = SUM(**Item**.value)

	A
1	Income
2	Item
3	33
4	44
5	20
6	Total
7	97

Functional Dependencies

- *Functional Dependency* (FD): A → B
- We compute the business logic from the data, by inferring FDs
- They are the building blocks for constructing ClassSheet (CS) models

Some Examples of FDs

◇	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	**com_code**	**upc**	**description**	**size**	**case**	**nitem**	**store**	**week**	**move**	**qty**	**price**	**onsale**	**profit**	**ok**
2	653	1111140009	DOVE DISH LIQUID	42 OZ	9	2851281	100	383	16	1	2.19		7.01	1
3	653	1111140009	DOVE DISH LIQUID	42 OZ	9	2851281	100	384	7	1	2.19		3.07	1
4	653	1111140009	DOVE DISH LIQUID	42 OZ	9	2851281	100	385	15	1	2.19		6.57	1
5	...	...	...	...	...	...	...	...	...	...	...	...	...	...
6	654	1111165003	SUNLIGHT AUTO GEL	88 OZ	6	2857061	100	390	6	1	3.75		4.50	1
7	654	1111165003	SUNLIGHT AUTO GEL	88 OZ	6	2857061	100	391	11	1	3.39	S	7.46	1
8	654	1111165003	SUNLIGHT AUTO GEL	88 OZ	6	2857061	100	392	8	1	3.75		6.00	1
9	...	...	...	...	...	...	...	...	...	...	...	...	...	...

- com_code → upc, description
- size → upc
- nitem ↛ week

Inferring FDs

- We use a data mining algorithm to infer all FDs
- These algoithm produce too many FDs
- We use some heuristics to filter the “accidental” FDs

The Heuristics

Label semantics: usually keys are labeled "code" or "id"

Label arrangement: we prefer FDs respecting the order of columns

Antecedent size: small keys are preferable

Ratio: small antecedents and big consequents

Single value columns: columns always with the same value are too intrusive

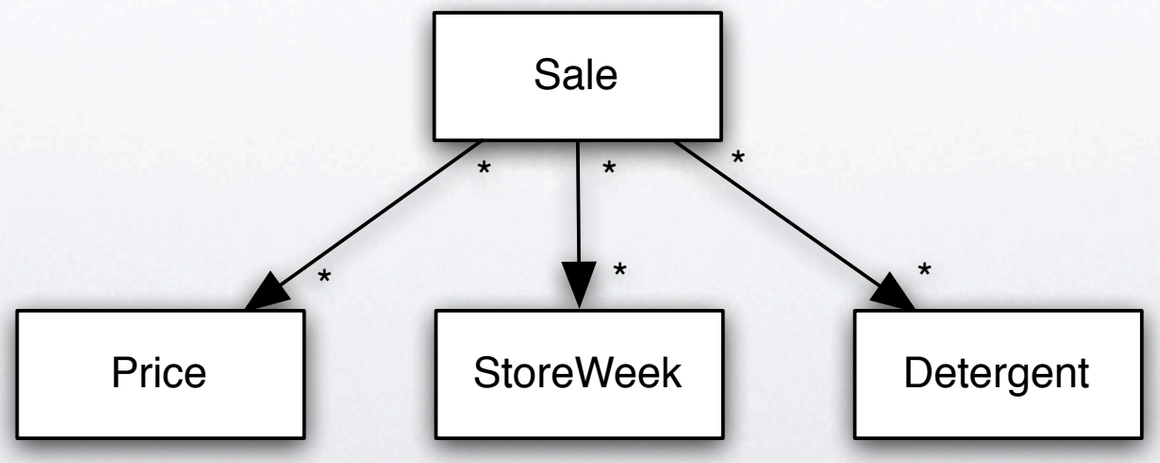
Foreign Keys

- Set of columns referencing other columns in some table
- Necessary for finding relationships

Candidate Key Att.		Foreign Key Att.	
Schema	**Attribute**	**Schema**	**Attribute**
Price	price	Sale	price
Detergent	upc	Sale	upc
StoreWeek	store	Sale	store
StoreWeek	week	Sale	week
StoreWeek	profit	Sale	profit

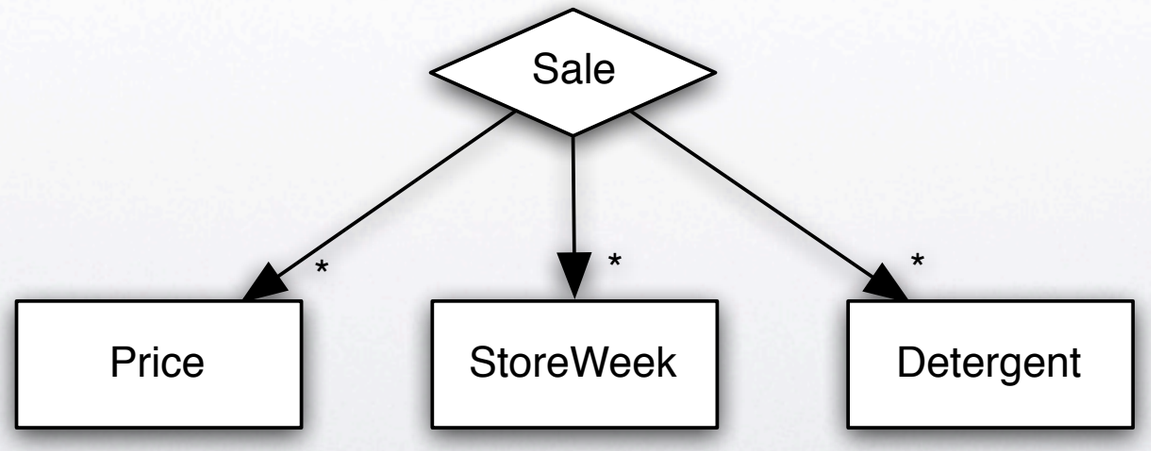
Relational Intermediate Graph

- Graph containing the entities and the foreign keys
- Four entities all connected to “Sale”



Detecting Relationships

- "Sale" is a relationship because all its columns are FKs to other table



Generating ClassSheets

- A (A1,...,An), and default values da1,...,dan

	A		
1	A		
2	A_1	...	A_n
3	a_1=da_1	...	a_n=da_n
⋮			

- Two entities with a FK: the FK columns have references to the other table

Generate Relationships

M (M1,...,Mr, Mr+1,...,Ms)

N (N1,...,Nt, Nt+1,...,Nu)

R (M1,...,Mr, N1,..., Nt, R1,...,Rx, Rx+1,...,Ry)

	A									...
1	**R**			**Mkey**						
2				M_1	...	M_r	R_1	...	R_x	
3				$m_1 = M.M_1$	...	$m_r = M.M_r$	$r_1 = dr_1$	...	$r_x = dr_x$	
4	**Nkey**									
5	N_1	...	N_t	R_{x+1}	...	R_y				
6	$n_1 = N.N_1$	...	$n_t = N.N_t$	$r_{x+1} = dr_{x+1}$	...	$r_y = dr_y$				
⋮										
7										
8	**A**									
9	**N**									
10	N_1	...	N_t	N_{t+1}	...	N_u				
11	$n_1 = dn_1$	...	$n_t = dn_t$	$n_{t+1} = dn_{t+1}$	...	$n_u = dn_u$				
⋮										
12										
13	**A**									
14	**M**									
15	M_1	...	M_r	M_{r+1}	...	M_s				
16	$m_1 = dm_1$	...	$m_r = dm_r$	$m_{r+1} = dm_{r+1}$	...	$m_s = dm_s$				
⋮										

Special Case 1

M (M1,...,Mr, Mr+1,...,Ms)

N (N1,...,Nt, Nt+1,...,Nu)

R (M1,...,Mr, N1,...,Nt, R1,...,Rx)

	A						...
1	**R**			**Mkey**			
2				M_1	...	M_r	
3				$m_1 = M.M_1$	...	$m_r = M.M_r$	
4	**Nkey**						
5	N_1	...	N_t	R_1	...	R_x	
6	$n_1 = N.N_1$	...	$n_t = N.N_t$	$r_1 = dr_1$	...	$r_x = dr_x$	
⋮							

Special Case II

M - Detergent

N - StoreWeek

R - Sale

	A	B	C	D	E	F	G	...
1	**Sale**			**DishKey**				
2				**Upc**	**Qty**	**Price**	**Ok**	
3				upc=**Detergent.Upc**	qty=0	price=**Price.Price**	ok=""	
4	**StoreWeek**							
5	**store**	**week**	**profit**	**Move**				
6	store=""	week=""	profit=move* **Price.**price*0.2	move=0				
⋮								
7								

8	A	B	C	D	E	F
9	**Detergent**					
10	**Upc**	**Com_code**	**Description**	**Size**	**Case**	**Nitem**
11	upc=""	com_code=""	description=""	size=0	case=0	nitem=""
⋮						
12						

13	A	B
14	**Price**	
15	**Price**	**Onsale**
16	price=0	onsale=""
⋮		

Special Case III

For cases with more than three classes:

- M and N should have small keys
- The number of empty cells created by the cell class should be minimal

Evaluation

- 27 spreadsheets, with a total of 121 worksheets (from the book *The Art of Modeling with Spreadsheets)*
- 66 out of the 121 contain formulas

Research Questions

RQ1: In how many cases is ClassSheet inference applicable?

RQ2: How many of the table and relationship structures that can be identified in the data can be successfully captured by ClassSheets inferred by our tool?

RQ3: In which cases does ClassSheet inference fail?

Results

Run the algorithms and visually evaluated the results:

- Inferred tables/relations: 176
- Failed: 13
- *Bad* results: 12
- *Acceptable* results: 27
- *Good* results: 124

Discussion

- We can produce ClassSheet models for 92% of the existing tables
- Of these, 76% are classified as good
- The failures occurred because no structure was found or FDs not in the data
- The results are encouraging, but could be improved by external information (more FDs or headers)

HaExcel

- Algorithms defined as reusable Haskell libraries
- Binding from Excel to Haskell
- Gumeric Haskell front-end

Conclusions

- We believe that models can improve spreadsheet usage (paper in preparation)
- But the usual lack of models is a problem
- We presented a technique to infer ClassSheet models from legacy spreadsheets
- The results are encouraging, but the technique could benefit from external info (more FDs or headers)