EMPOWERING THE USE OF VARIANT TABLES IN MASS CUSTOMIZATION

Product Management Haag GmbH

Albert Haag and Laura Haag

Combined slides for the MCP-CE (Novi Sad) and CONFWS'18 (Graz), September 2018

PMH stands for Product Management Haag GmbH

Mass customization is about personalized products like T-shirts and cars. We have combined the slides of the two related presentations that were held within a week of each other.

QUASI-FINITE DOMAINS

DEALING WITH THE INFINITE IN MASS CUSTOMIZATION

Product Management Haag GmbH

Albert Haag and Laura Haag

Combined slides for the MCP-CE (Novi Sad) and CONFWS'18 (Graz), September 2018

PMH stands for Product Management Haag GmbH

Mass customization is about personalized products like T-shirts and cars. We have combined the slides of the two related presentations that were held within a week of each other.

- ★ *Very large tables (via compression)*
- ★ *Very fast evaluation of compressed form*
- ★ *vBase18™ Variant Management System*
- ★ *Substantially enhance legacy configurators (e.g. SAP VC)*
- ★ *Mass Customization (MC) is a natural application*

☆ Leitmotiv: 85% of a product model might be pure tables

Premises Concerning MC:

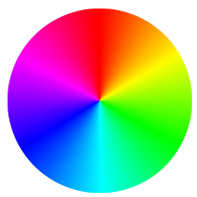
- Many mass customizable products can be defined by their variants
- Compression can handle the combinatorial explosion of variants

Simple compression to “c-tuples” already goes a long way

Mass Customization (MC)

- MC = mass production + individualization

Ford Model-T 1909	Porsche 911 1989	BMW 2029?
One color “Black”	Small list of colors	Any RGB color 16777216 colors



Picture by ModelTMitch - Own work, CC BY-SA 4.0, WikiPedia link

Quote attributed to Henry Ford (1909): “A customer can have a car painted any color he wants as long as it’s black”

In 1909 mass production was the innovation more important than the individualization then available through traditional artisans

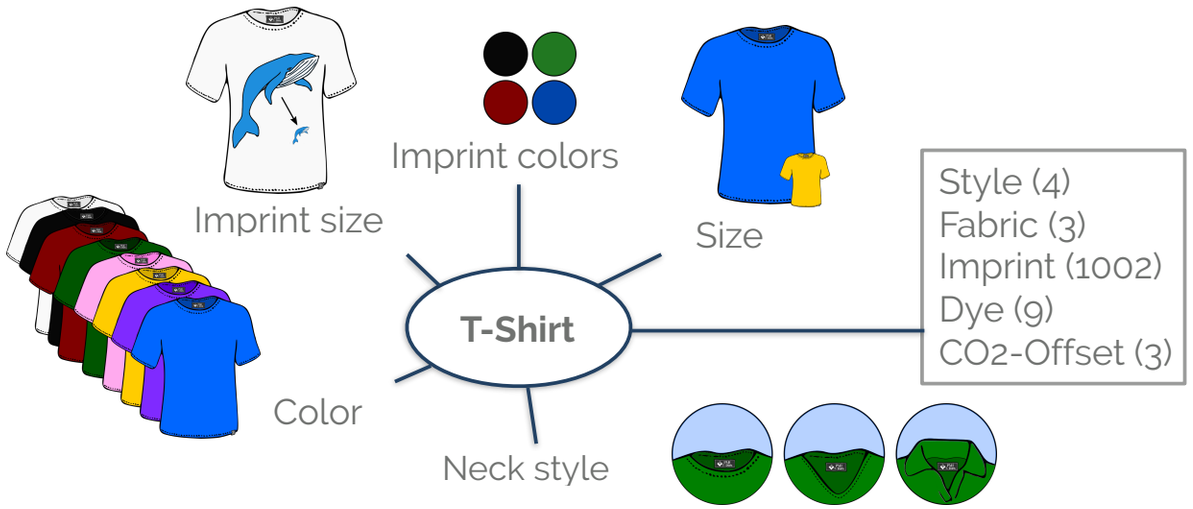
Key Idea of Product Variants

- One generic product
- Variants distinguished by finite set of additional descriptive properties

- Simple example: felt pens mass produced in three colors
  - Product data and bill of materials maintained once
  - One descriptive property, *Color*, distinguishes variants

MC T-Shirt - More than 270 Million Variants

10 properties, 1044 features, > 270 million combinations



- ★ Number of variants increases exponentially with choices
- ★ But not the complexity

MC T-shirt is the generic product

Simple T-Shirt – Eleven Variants in a Table

Sizes M; L Sizes S; M; L

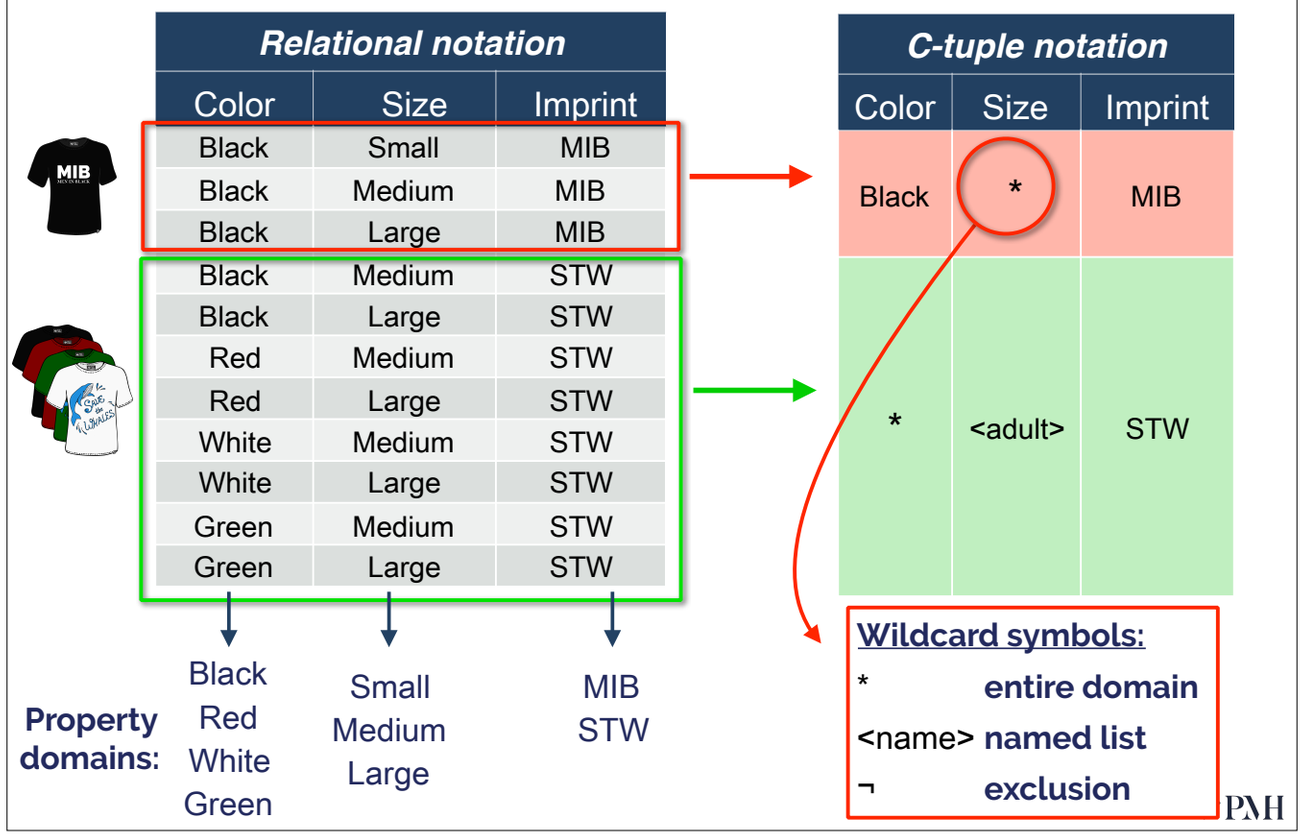
Two “vintage” imprints

Variant Table for Simple T-Shirt		
Color	Size	Impprint
Black	Small	MIB (Men in Black)
Black	Medium	MIB
Black	Large	MIB
Black	Medium	STW (Save the Whales)
Black	Large	STW
Red	Medium	STW
Red	Large	STW
White	Medium	STW
White	Large	STW
Green	Medium	STW
Green	Large	STW

⭐ Variant tables define legal combinations (function both as constraints and database tables)

Our terminology: product property: the “observable”; product feature is a value assignment to a property

A C-Tuple Expresses Multiple Combinations



The first c-tuple with reddish background is interpreted as “black T-shirts come in any size with imprint ‘MIB’”. The second c-tuple with greenish background is interpreted as “T-shirts with imprint ‘STW’ come in any <adult> size (meaning ‘Medium’ or ‘Large’ – that is ¬‘Small’) and in any color”

Configuration related filtering queries of variant tables:

- Result set of valid variants
- Domain restrictions
- Local (constraint) propagation

The following slides illustrate these queries

Filtering Query

Relational notation		
Color	Size	Imprint
Black	Small	MIB
Black	Medium	MIB
Black	Large	MIB
Black	Medium	STW
Black	Large	STW
Red	Medium	STW
Red	Large	STW
White	Medium	STW
White	Large	STW
Green	Medium	STW
Green	Large	STW

Query c-tuple		
Color	Size	Imprint
Red	*	*

Result set

SELECT * FROM Simple-T-shirt WHERE <Query c-tuple>

Query conditions are c-tuples

Domain Restriction for Color

Relational notation		
Color	Size	Imprint
Black	Small	MIB
Black	Medium	MIB
Black	Large	MIB
Black	Medium	STW
Black	Large	STW
Red	Medium	STW
Red	Large	STW
White	Medium	STW
White	Large	STW
Green	Medium	STW
Green	Large	STW

Query c-tuple		
Color	Size	Imprint
Red	*	*

Black
Red
White
Green

SELECT DISTINCT Color FROM Simple-T-shirt WHERE <Query c-tuple>

Domain Restriction for Size

Relational notation		
Color	Size	Imprint
Black	Small	MIB
Black	Medium	MIB
Black	Large	MIB
Black	Medium	STW
Black	Large	STW
Red	Medium	STW
Red	Large	STW
White	Medium	STW
White	Large	STW
Green	Medium	STW
Green	Large	STW

Query c-tuple		
Color	Size	Imprint
Red	*	*

Small
Medium
Large

SELECT DISTINCT Size FROM Simple-T-shirt WHERE <Query c-tuple>

Domain Restriction for Imprint

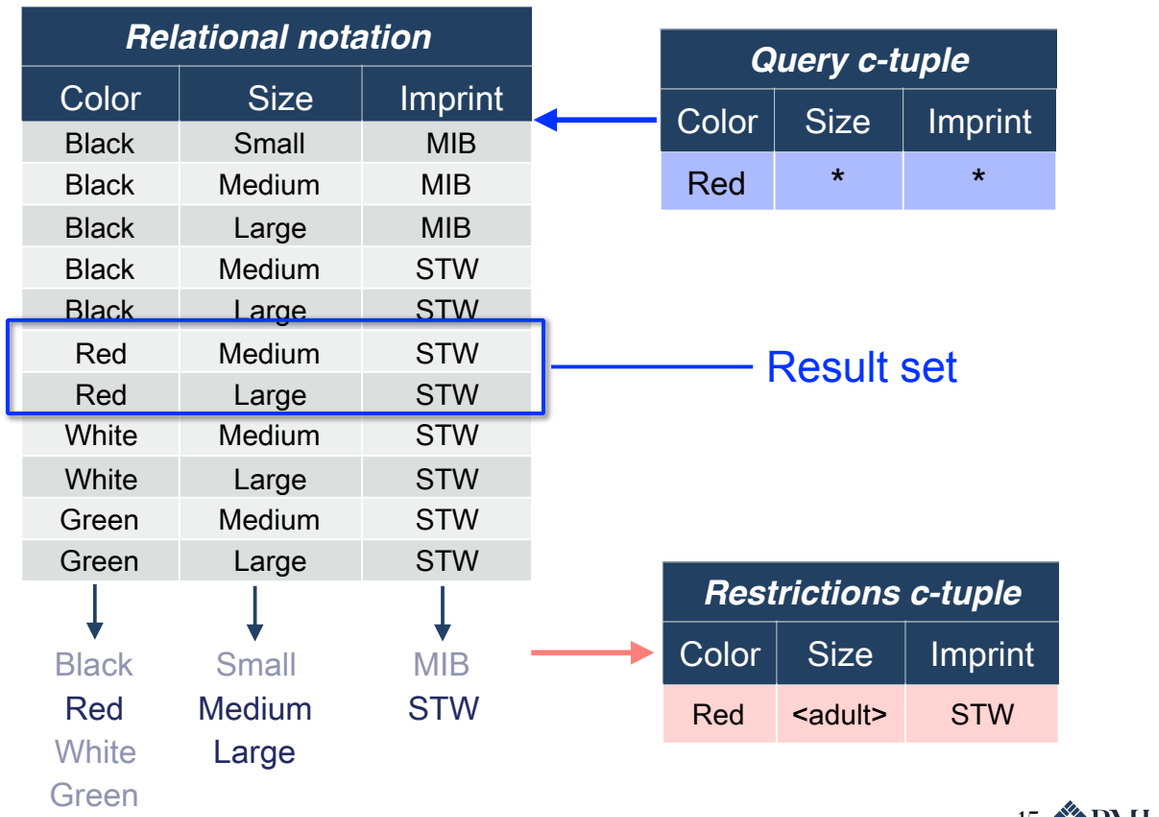
Relational notation		
Color	Size	Imprint
Black	Small	MIB
Black	Medium	MIB
Black	Large	MIB
Black	Medium	STW
Black	Large	STW
Red	Medium	STW
Red	Large	STW
White	Medium	STW
White	Large	STW
Green	Medium	STW
Green	Large	STW

Query c-tuple		
Color	Size	Imprint
Red	*	*

MIB
STW

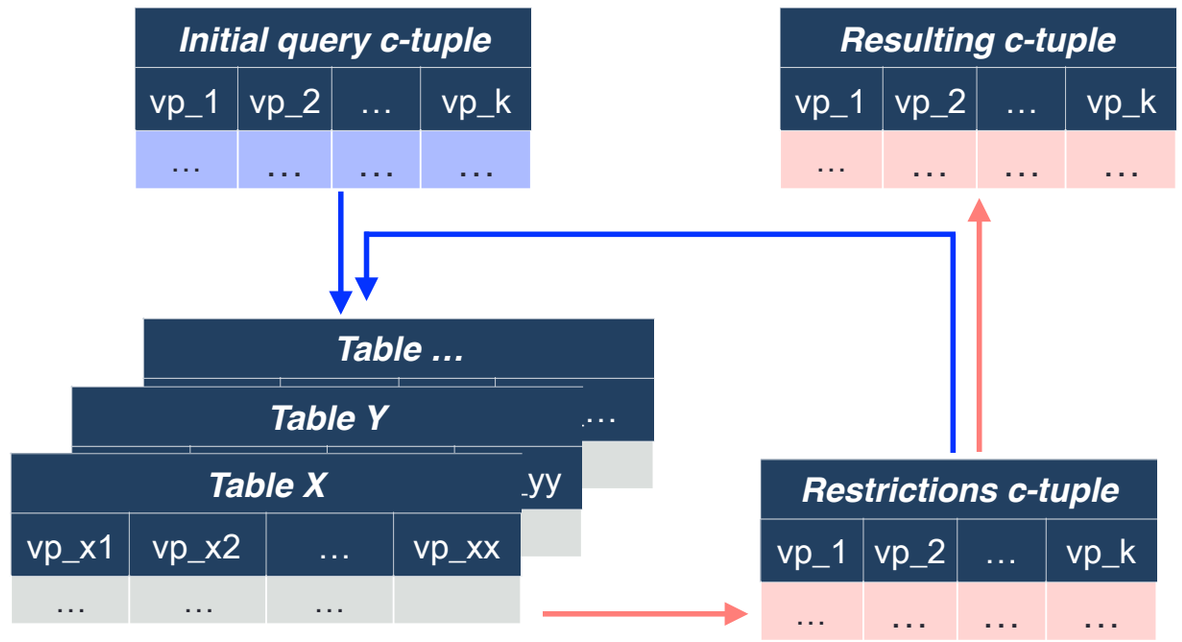
SELECT DISTINCT Imprint FROM Simple-T-shirt WHERE <Query c-tuple>

Domain Restrictions as C-Tuple



One SELECT DISTINCT query for each of the columns can be combined to yield one resulting c-tuple representing the resulting restriction used in constraint propagation (arc-consistency)

Local Constraint Propagation



☆ Apply restrictions to all tables until no further restriction is possible (arc-consistency: result)

The query c-tuple is applied to all tables. Each ensuing restriction of the c-tuple is again applied to all tables, until no further restrictions occur. The c-tuple that allow no further restriction is the resulting overall domain restriction. This is a state of “arc-consistency”)

Demo – MC T-shirt as two tables:

- 241 Million Sales variants
- 276 Million Manufacturing Variants

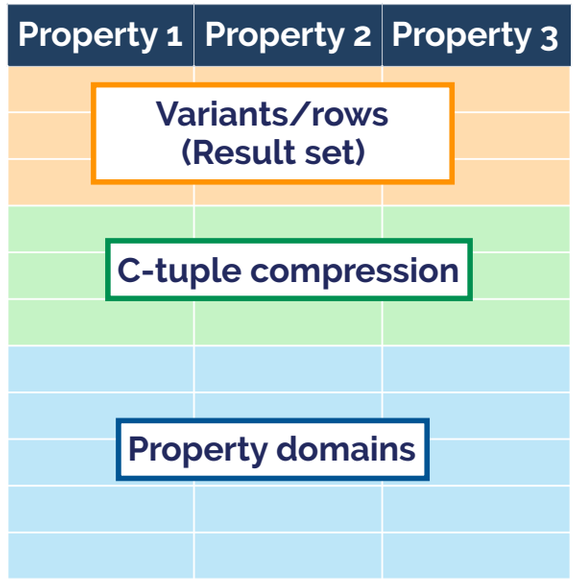
For purposes of demonstration, we have defined both a sales and a manufacturing model. Local propagation can be used to ensure consistency of the two models with each other.

Four MC T-Shirt Constraints

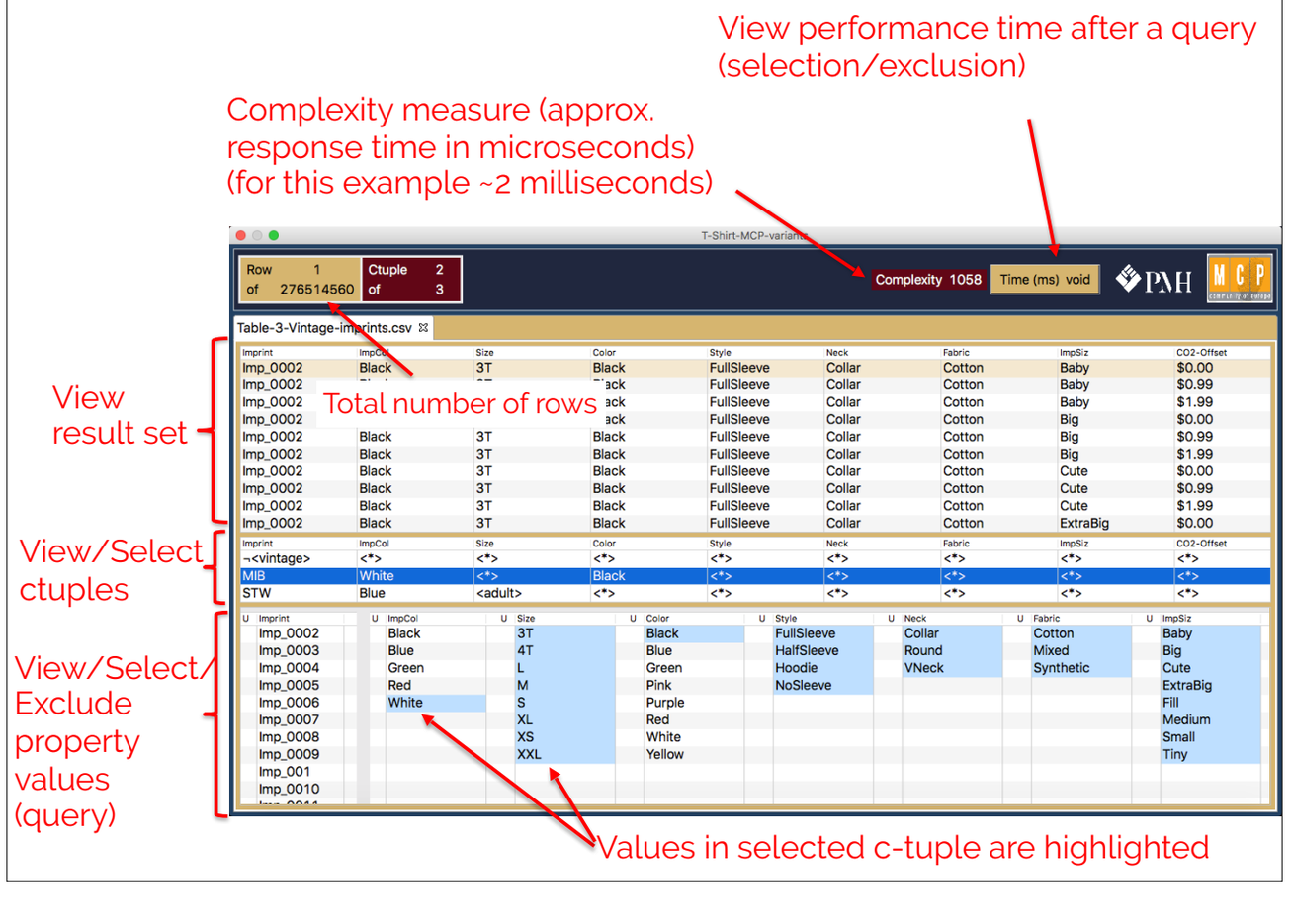
- Imprint MIB: Black t-shirt, white imprint
- Imprint STW: Blue imprint, adult sizes
- Imprint color ≠ T-shirt color (Sales)
- Fabric ⬌ Color ⬌ Dye (Manufacturing)

The first two constraints are common to both sales and manufacturing. Sales cares about imprints being clearly distinguishable. Manufacturing does not care about this, but cares about the dye needed for the T-shirt.

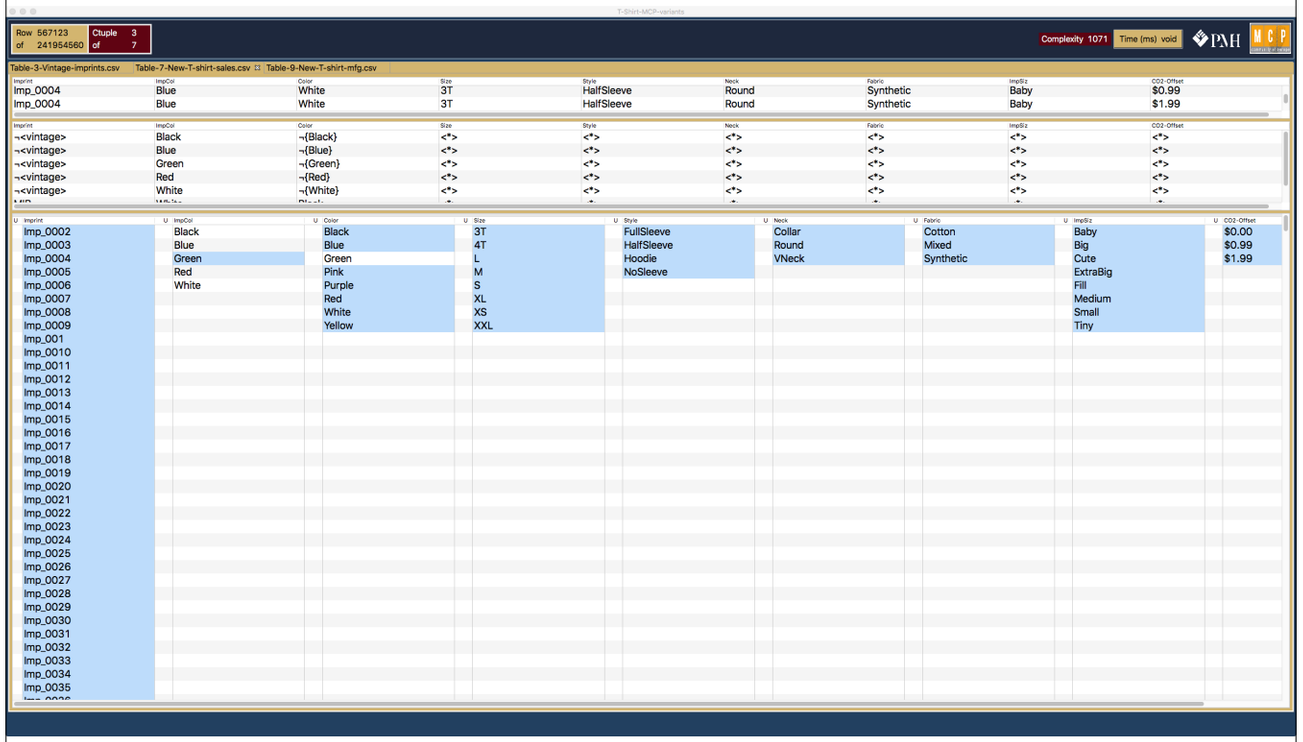
Table View in Demo



MC T-shirt model – overall variant table

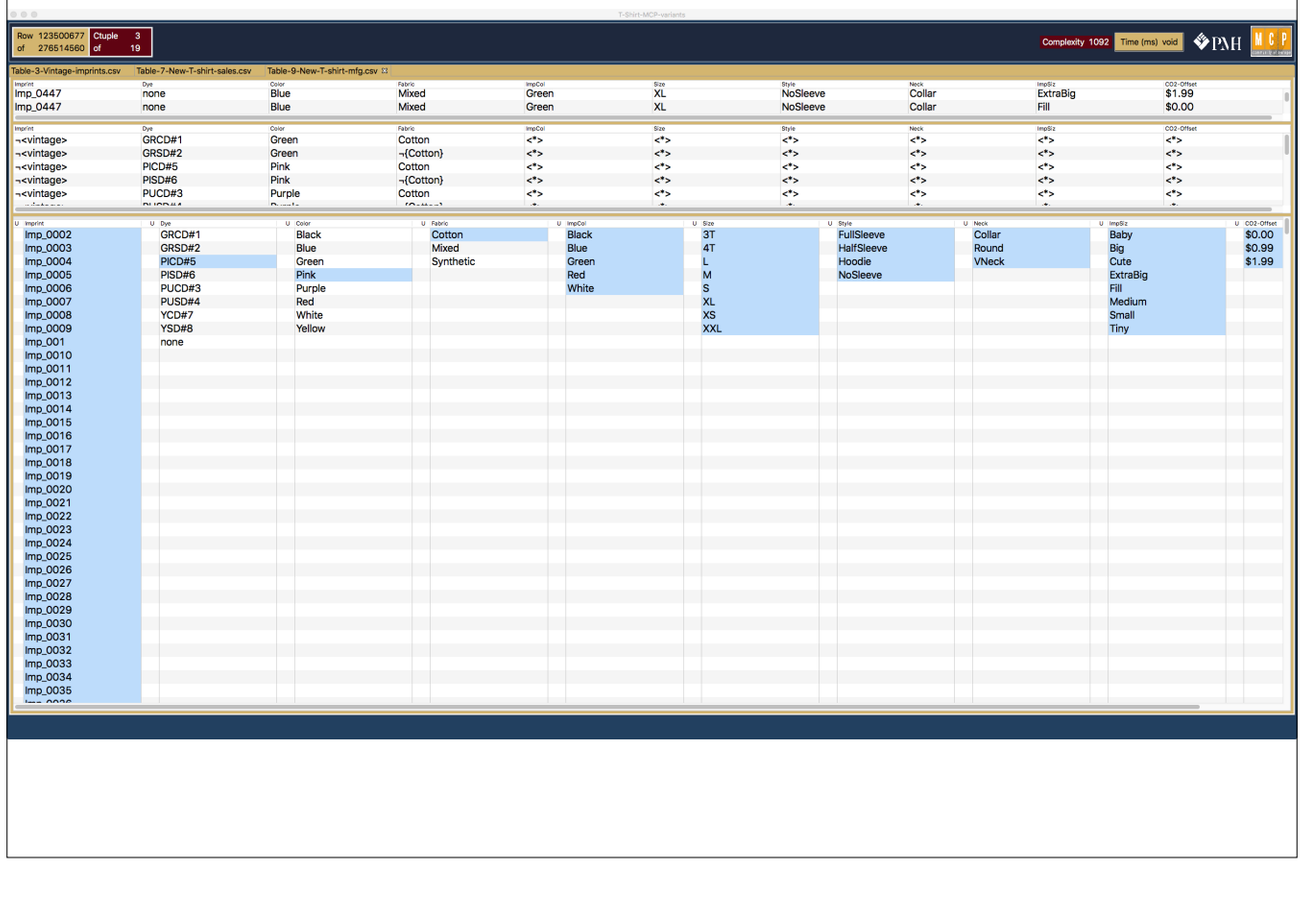


MC T-shirt sales model



Result set scrolled to row 567123 of 241954560 sales variants; c-tuple viewer selected third of seven c-tuples; the values in this c-tuple are highlighted in the domains at the bottom; no selections or exclusions have been done.

MC T-shirt manufacturing model

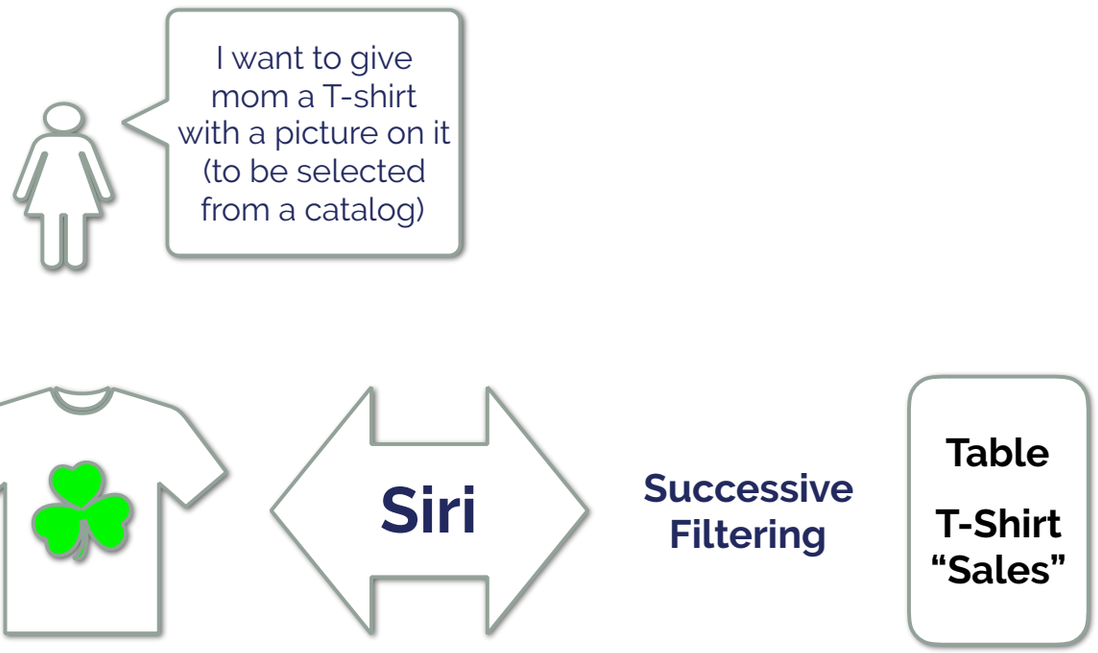


Result set scrolled to row 123500677 of 276514560 manufacturing variants;
c-tuple viewer selected third of 29 c-tuples; the values in this c-tuple are highlighted in the domains at the bottom

Criticism of MC T-shirt with finite domains:

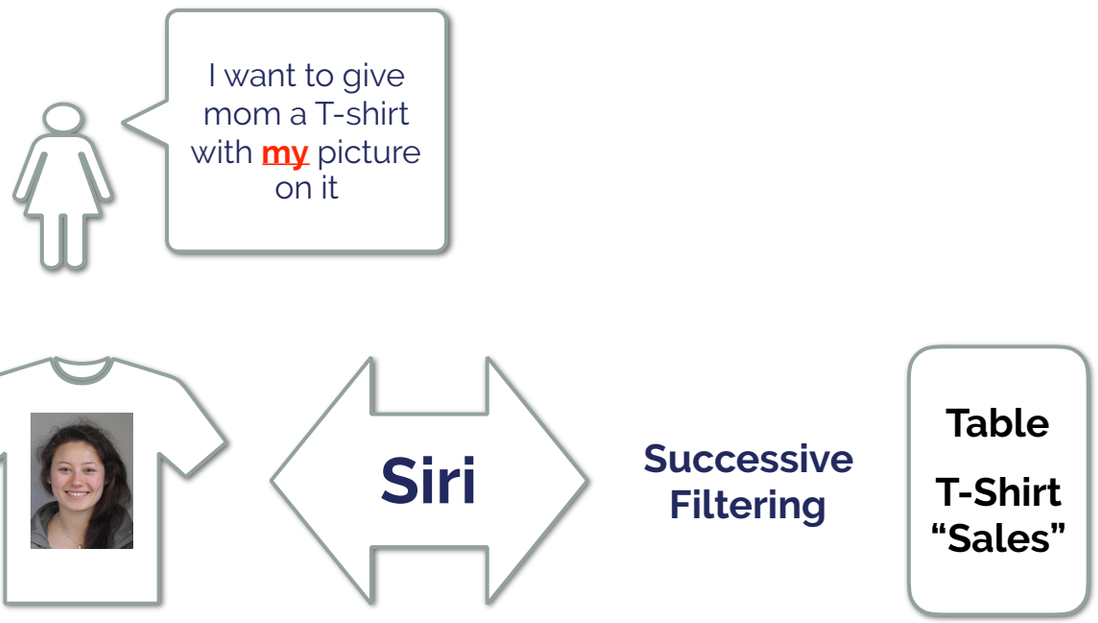
We should be able to deal with “additional values” for “Imprint”

Finite MC Scenario for Ordering a T-shirt



If Siri can order a Taxi, she can order a T-shirt. All it takes is finding a supplier, understanding their nine relevant T-shirt properties, and querying the sales variant table, which is in essence a "product catalog"

Modern MC Scenario for Ordering a T-shirt



Here there is a slight additional complication. Siri needs to understand which filenames are legal. An illegal file name should lead to an appropriate error message, something we have not discussed here.

Idea: Introduce quasi-finite (QF)-symbols to encode infinite sets

Tables and domains remain finite in terms of symbols, but some symbols express infinite sets

Goals:

- Support compression as before
- Support constraint propagation/solving as before

Allow Personal Image in Variant Tables

- Extend both sales and manufacturing table with the one c-tuple below
  - **<img-filename>** is a QF-symbol for the regular expression: ***.jpg | *.tif** to be allowed for Imprint
  - Column *Dye* only relevant for manufacturing table

Imprint	*ImpCol*	*Size*	*Color*	*Style*	*Neck*	*Fabric*	*ImpSiz*	*CO2-Offset*	*Dye*
<img-filename>	*	*	White	*	*	*	*	*	none

Sample Interaction with QF-Symbols

Using QF-symbols for real-valued intervals

QF-Symbols in Domain Restriction

Query for all small ("S") T-shirts

Size	ImpSize	Scale
S	Cute	[0.1, 0.2]
S	Big	[0.5, 0.6]
S	Fill	[0.7, 0.9]
M	Cute	[0.2, 0.3]
M	Big	[0.75, 0.9]
M	Fill	[1.0, 1.2]
L	Cute	[0.25, 0.4]
L	Big	[0.75, 0.9]
L	Fill	[1.3, 1.9]

Query c-tuple		
Size	ImpSiz	Scale
S	*	*

Restrictions c-tuple		
Size	ImpSize	Scale
S	*	[0.1, 0.2], [0.5, 0.6], [0.7, 0.9]

"Xnumeric"

Xnumeric datatype: an ordered set of normalized (i.e. non-overlapping) intervals, including singleton intervals to represent individual numbers

Normalized QF-Symbols in Domain Restriction

Query: All "Cute" imprint sizes

Size	ImpSize	Scale
S	Cute	[0.1, 0.2]
S	Big	[0.5, 0.6]
S	Fill	[0.7, 0.9]
M	Cute	[0.2, 0.3]
M	Big	[0.75, 0.9]
M	Fill	[1.0, 1.2]
L	Cute	[0.25, 0.4]
L	Big	[0.75, 0.9]
L	Fill	[1.3, 1.9]

Query c-tuple

Size	ImpSiz	Scale
*	Cute	*

Restrictions c-tuple

Size	ImpSize	Scale
*	Cute	[0.1, 0.4]

"QF-interval"
(xnumeric normalization)

[0.1, 0.2]
[0.2, 0.3]
[0.25, 0.4]
} [0.1, 0.4]

Normalized xnumeric datatype: merge overlapping intervals to one interval

QF-Symbols in Query and Condition

Query: Scale IN [0.6, 1.1]

Size	ImpSize	Scale
S	Cute	[0.1, 0.2]
S	Big	[0.5, 0.6]
S	Fill	[0.7, 0.9]
M	Cute	[0.2, 0.3]
M	Big	[0.75, 0.9]
M	Fill	[1.0, 1.2]
L	Cute	[0.25, 0.4]
L	Big	[0.75, 0.9]
L	Fill	[1.3, 1.9]

Query c-tuple		
Size	ImpSiz	Scale
*	*	[0.6, 1.1]

Restrictions c-tuple		
Size	ImpSize	Scale
*	Big;Fill	0.6, [0.7, 0.9], [1.0, 1.1]

(xnumeric intersection with normalization)

Summary: Necessary QF-Operations

- We need following operations given two QF-symbols for the same product property
  - **Union** (normalization)
  - **Intersection** (specialization)
  - **Negation** (good to have, needed in VDD construction)
    - Implies exclusion sets (see paper))

⭐ Any set of symbols that satisfy these requirements can be used for QF-symbols, e.g. “Rectangles” etc.

⭐ Generalizes to a “specialization relation”

Specialization Relation

A partial ordering on a set of symbols (QF-symbols or values) such that:

- *Constraint processing* need not consider an otherwise valid symbol assignment to a product property in the presence of a more special one (*procedural-subsumption property*).
- A symbol assignment is logically implied by any of its specializations (*semantic-compatibility property*).
- Negation inverts specialization (*symmetry-under-negation property*).

References:

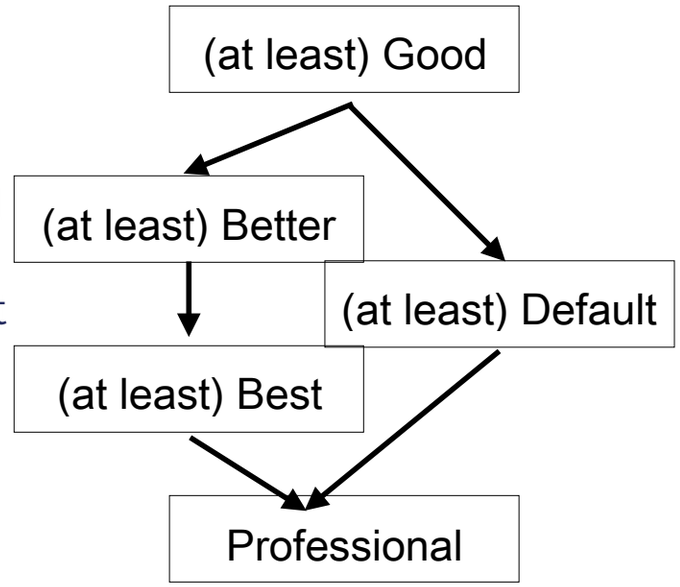
- Albert Haag and Laura Haag, ‘Empowering the use of variant tables in mass customization’, in Proceedings of the MCP-CE 2018 conference, Novi Sad, Serbia, September 19-21, 2018., pp. 180-189, (2018).
- Albert Haag, ‘Quasi-finite domains: relaxing finiteness of domains and mass customization’, in Proceedings of the 20th International Configuration Workshop, Graz, Austria, September 27-28, 2018., pp. 77-84, (2018).

Examples of Specialization

- An additional value for Property *Imprint*: specializes **<img-filename>** if it matches the regular expression; otherwise it signals a contradiction
  - "my-image.jpg" is a specialization
  - "blueberry" is an illegal assignment
- If the two assignments **Scale IN [0.6, 1.1]** and **Scale IN [1.0, 1.2]** need to be considered simultaneously, they must be replaced by their common specialization **Scale IN [1.0, 1.1]**
- Next slide illustrates specialization for an additional property (image) *Resolution*, with a finite domain arranged in a specialization relation

Image Resolution — Finite Specialization Relation

- If the two assignments **Resolution = 'Better'** and **Resolution = Default'** need to be considered simultaneously, they must be replaced by their common specialization **Resolution = 'Professional'**



★ Example of finite specialization relation

- Email: info@product-management-haag.de
- Web: http://product-management-haag.de
- www.vbase18.com

ANY QUESTIONS?

THANK YOU!

We have dedicated the vBase18-engine to our son and brother Olaf “in memoriam”. He was in this at the start and we sorely miss him.