Categorising Anticipatory Systems

Dimitrios Sisiaridis, Nick Rossiter & Michael Heather

Northumbria University, Newcastle NE2 1XE, UK

Symposium 2: Applied Mathematics, Dynamical Systems, Logics and Category Theory

CASYS'11 - International Conference on COMPUTING ANTICIPATORY SYSTEMS
HEC Management School
University of Liege, LIEGE, Belgium, August 8-13, 2011

Aristotle

- Aristotle coined the word 'categories' to describe a structure of classes
  - Used as the title of one of the books of his treatise on logic, the *Organon*.
- This begun a 2,000 year history for the category to describe the structured level at the foundation of logic.

Symbolic Logic Diversion

- The study of logic diverted to the symbolic logic of set theory around 1900
  - So the concept of a category with classes at various levels was no longer easy to represent.
  - For the elements for the set are independent of one another and a set cannot be a member of itself
  - No inherent possibility to represent recursion nor relations other than by external functions.

Indispensable Categories

- Nevertheless the category has become an indispensable component for many disciplines and the concept is still developing today.
- It is the primary classification system for Wikipedia which itself has about 500 types of categories defined.
- With advances in information systems the concept of typing is an aspect of categories that has increased in importance.

Types of Anticipation

- There are two fundamental types of anticipatory system
  - the strong
  - the weak
- which differ markedly one from the other.

Strong Anticipation

- The strong is a unique intension for any given configuration of the Universe.
- The strong anticipatory system is the real world one
    - needing some metaphysical representation or at least a view at the level of the world itself.
- Strong anticipation is not then a property of an anticipatory system.
- Rather it is the essence of the system itself and requires impredicative (self-referencing) mathematics.

Weak Anticipation

- Weak anticipation on the other hand is a property of an anticipatory system and extensionally degenerate for any physical reality.
- It is predicative and may therefore be modelled in set theory.

Isomorphism in Context

- This mirrors the distinction in the category theory of mathematics which represents reality up to some isomorphism.
- The strong version is up to natural isomorphism and formalises the concept of 'natural' as found in the real world.
- The weak is only up to some assumed isomorphism
    - For instance the category of sets holds up to the isomorphism of Zermelo-Fraenkel set theory with the axiom of choice.

Cartesian Closure for Strong Anticipation

- In category theory terms the anticipation of an anticipatory system resides in the relationships of its cartesian closed structure.
- That is the completeness of the whole
  - generalises the local completeness that Gödel proved for first order predicate logic of axiomatic systems.
- The cartesian closure provides the full formal rigour for strong anticipation.

Modelling for Strong Anticipation

- Weak anticipation models the strong.
- The subtypes of possible weak anticipatory systems can therefore be categorised by locally cartesian closed slice categories.
- These focus on material relationships for a particular context for the system.

Logical to Physical

- The underlying drive in knowledge engineering is to represent the real world effectively by mapping logical structures onto physical ones.
- Knowledge bases are created according to the activities and relationships in which we are interested.
- It is possible that there are more activities taking place between the identified entities, but the domain of interest is always the crucial factor.

Aim of Data Normalisation

- The normalization process, as a good knowledge engineering practice, aims to achieve good designs by testing relations for undesirable types of functional dependencies.
- Mathematical structures that match the physical world are both desirable and practicable for developing knowledge-bases and information systems in general.

Attempts with Sets

- Previous attempts at defining knowledge bases have involved set theory.
  - For example Ullman in 1988 in 70 pages of text developed a complex notation for representing many types of constraint such as functional, multivalued, join, generalised and inclusion.
  - Constraints were then developed based on these dependencies for the design of optimal data structures with respect to redundancy and update operations.

Problems with Sets

- The notation is
  - idiosyncratic, building on an intricate definition of collections of arrows
  - not a natural domain for set theory
  - external to implemented information systems.
- Leading to difficulties in understanding, in implementation and in enforcement.
- Such an approach is at the weak anticipation level.

Potential of Category Theory

- A more suitable form of mathematics is category theory where the fundamental structures are defined
    - in the form of arrows as identity object, category, functor, natural transformation and adjointness.
- In type theory Robert Seely in 1984 presented a proof that
    - the category LCCC (Locally Cartesian Closed Category) and
    - the category ML of syntactically presented Martin-Löf type theories (with Pi, Sigma, and extensional identity types)

Potential of LCCC

- LCCC do appear relevant for our work as they facilitate the construction of relationships in the context of the dependencies identified by Ullman.
    - Indeed LCCC, in the form of comma categories, can explain the relations in typed systems, as representations of logical structures that handle effectively the physical world.

Outline of Paper

- We next
  - define LCCC
  - provide further details on the various forms of normalization
  - bring together, with examples, the two strands of LCCC and normalization as pullbacks and pushouts
  - discuss the strengths and weaknesses of the methods of categorising data normalisation in the context of anticipatory systems

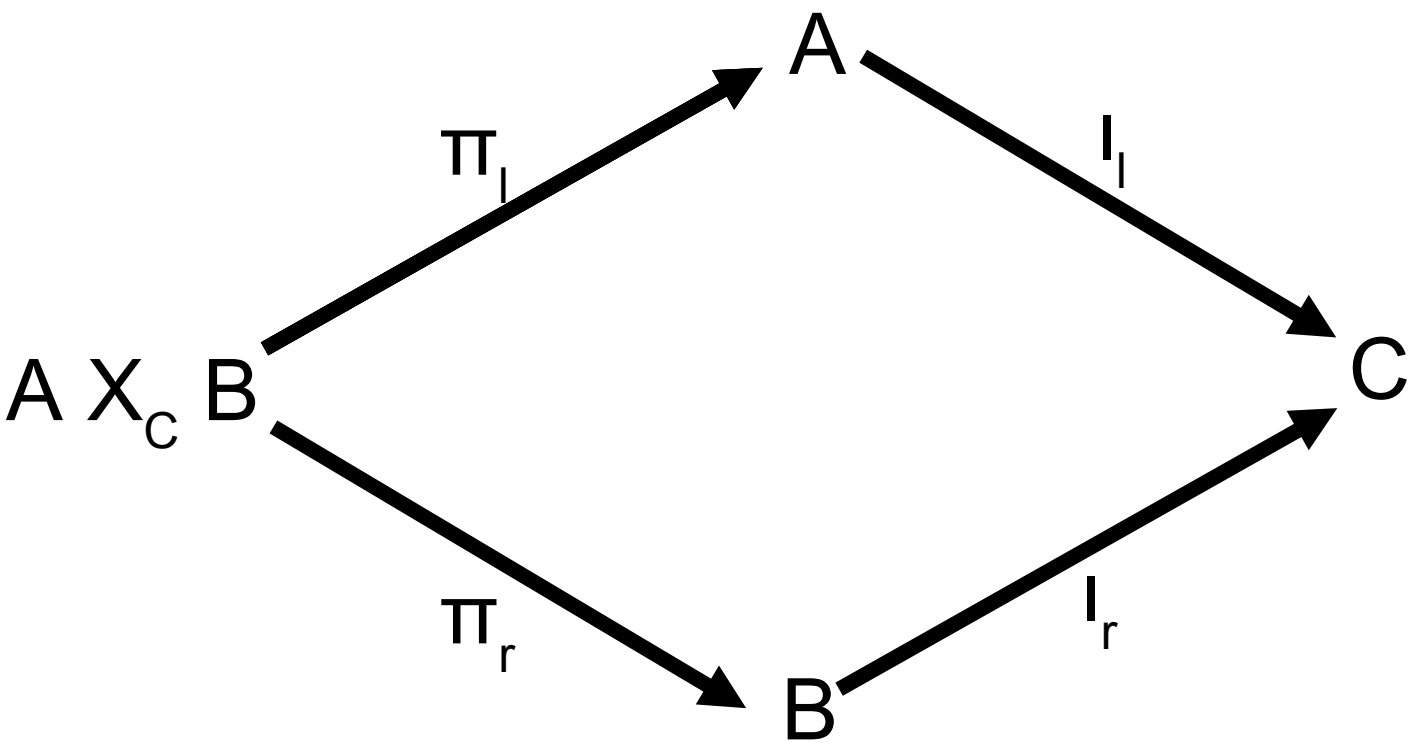
Formal Representation

- Based very much on
  - Cartesian closed category (CCC)
    - Connectivity (exponential)
    - Product (prerequisite for relationships)
    - Initial object (unique starting point)
    - Terminal object (unique finishing point)
  - Fits in with philosophy
    - Everything is connected
    - Everything is related
    - Everything is limited

LCCC

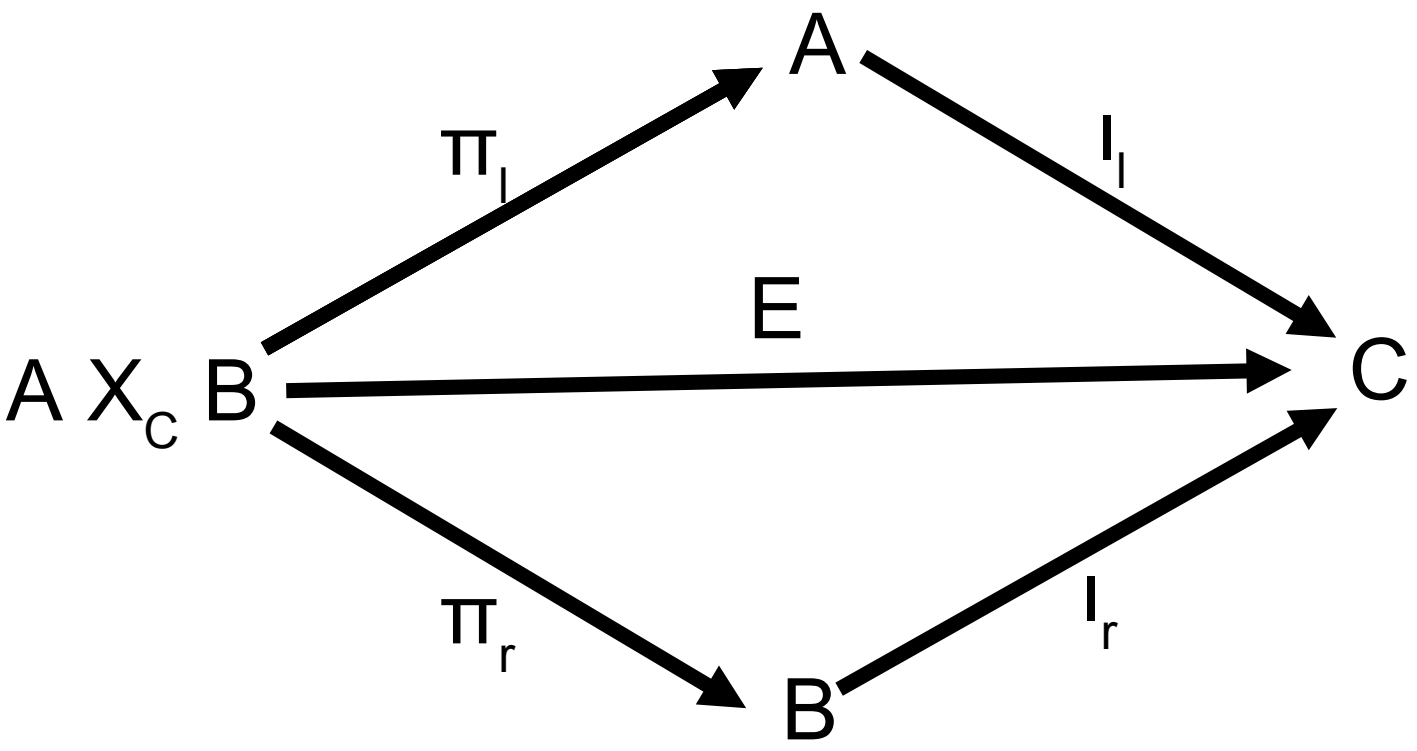
- In practice we use a variant of Cartesian closed categories
  - Locally Cartesian closed category
    - Product is replaced by a relationship
  - Product is all possible pairs
    - e.g. account number X borrower name (A X B)
  - Relationship is those pairs that satisfy a particular context
    - e.g. account number X borrower name in the context of cash owed (A X_C B)
  - In category theory this is a pullback (with adjointness properties)

Pullback



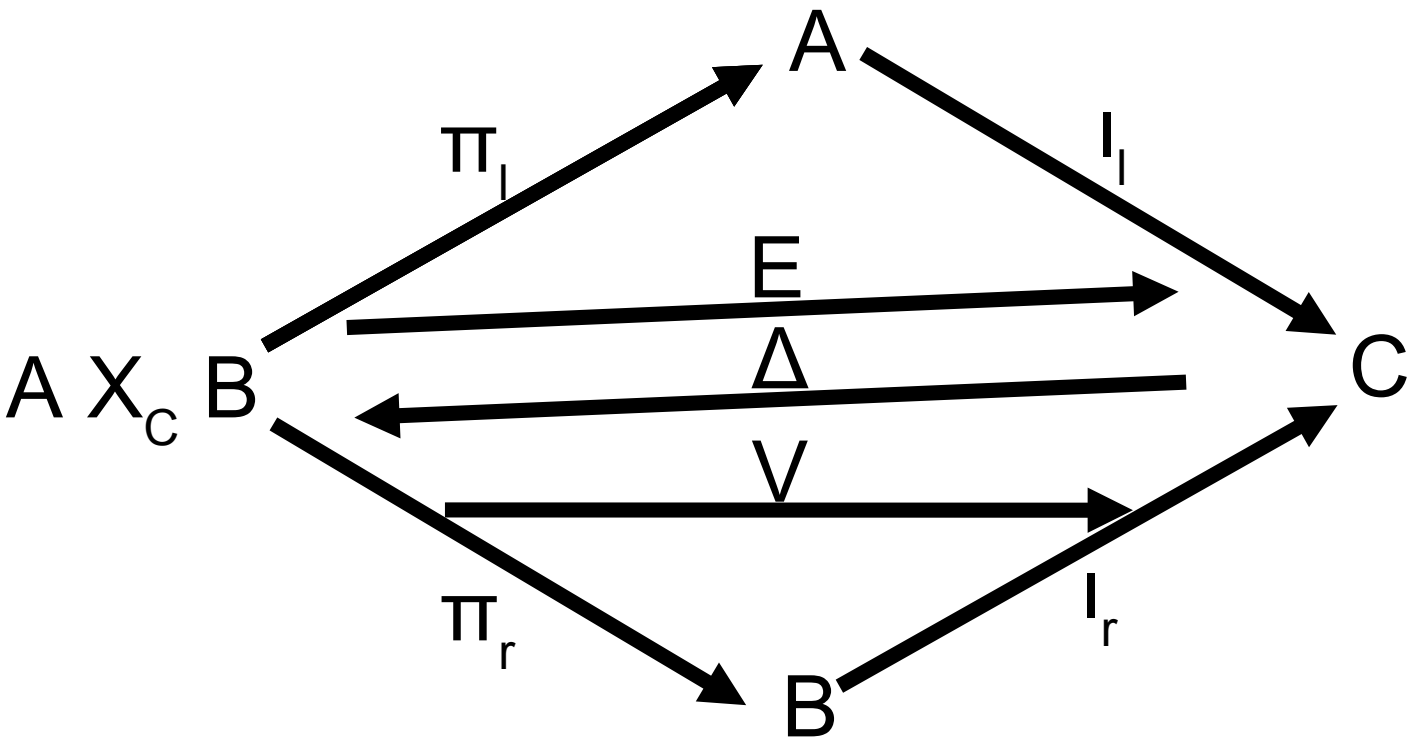
C is A+B+C

Pullback



E is an equaliser: $E = \iota_l \circ \pi_l = \iota_r \circ \pi_r$

Pullback



Adjointness requirements E ⊣ Δ and Δ ⊣ V

Working Assumption

- The Pullback has underpinned much of our work on information systems
- But is this justified?
- Information systems are open ended.
- We cannot prove all our instances of data are pullbacks.
- But we can try to relate pullbacks to accepted practice in software engineering.

Software Engineering Principles

- Information system data design
  - Normalisation
    - Commonly to 3NF (third normal form)
- How do these concepts relate to LCCC?
- LCCC have been popular in theoretical computing science
  - But little attempt to handle design issues

Normalisation Outline

- A relation comprises a collection of attributes
  - e.g. delivered (customer_id, customer_name, customer_address, item_code, driver_id, driver_name)
- Decide on those that provide uniqueness and make these the key
  - customer_id, item_code
- The others become non-key
  - customer _name, customer_address, driver_id, driver_name
- Requires knowledge of how things are done physically

Normalisation Stages

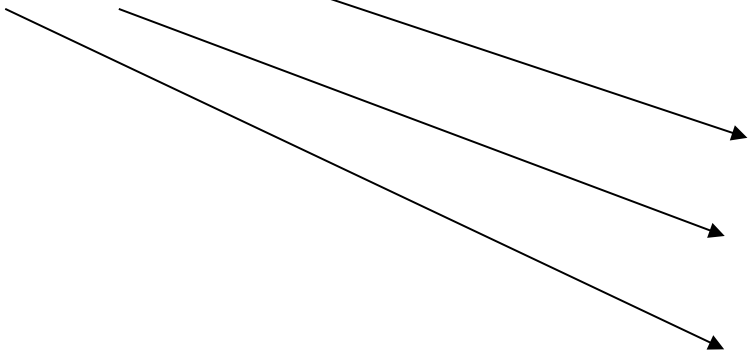
- Then check validity against 3 forms of increasing severity:
    - 1NF: for relation R each non-key attribute is functionally dependent on the key
    - 2NF: R is in 1NF and each non-key attribute is fully functionally dependent on the key (not dependent on any component of key)
    - 3NF: R is in 2NF and no non-key attribute is functionally dependent on another non-key attribute
- Maths in set theory is convoluted – students find it challenging. e.g. Ullman, J D, Principles of Database and Knowledge-base Systems (1988).
- Some category theory work has tried to directly represent set approach in categories – categorification

e.g. Johnson, M, & Rosebrugh, R, Sketch Data Models, Relational Scheme and Data Specifications, Electronic Notes in Theoretical Computer Science **61** 51-63 (2002).

1NF

- A relation is in 1NF if there is a functional dependency from the key to each non-key attribute.
- So expectation is:

customer_id, item_code → customer _name



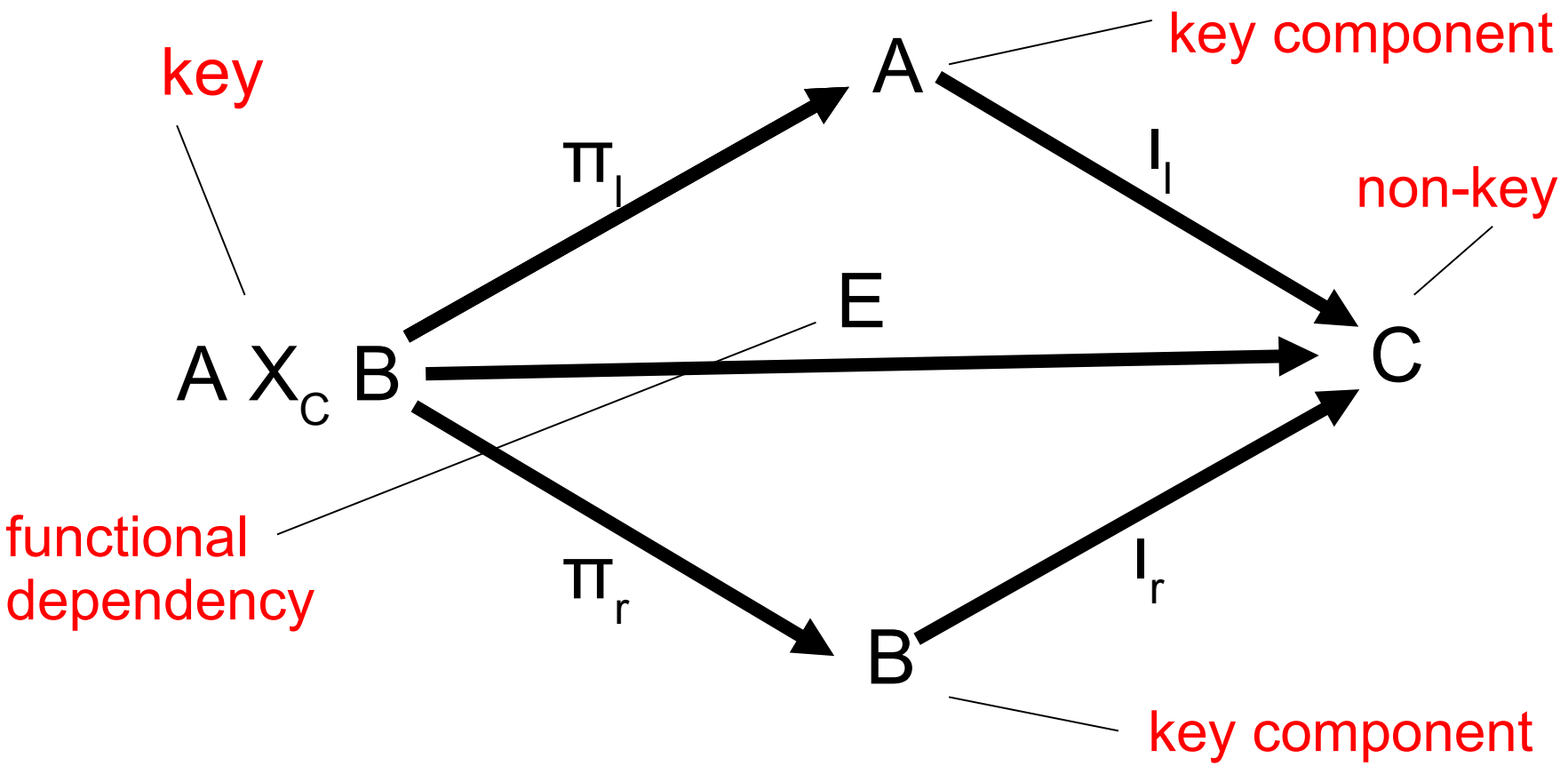
customer_address

driver_id

driver_name

If add something unrelated such as football_club then not in 1NF: need everything to be connected

LCCC view of 1NF - Pullback

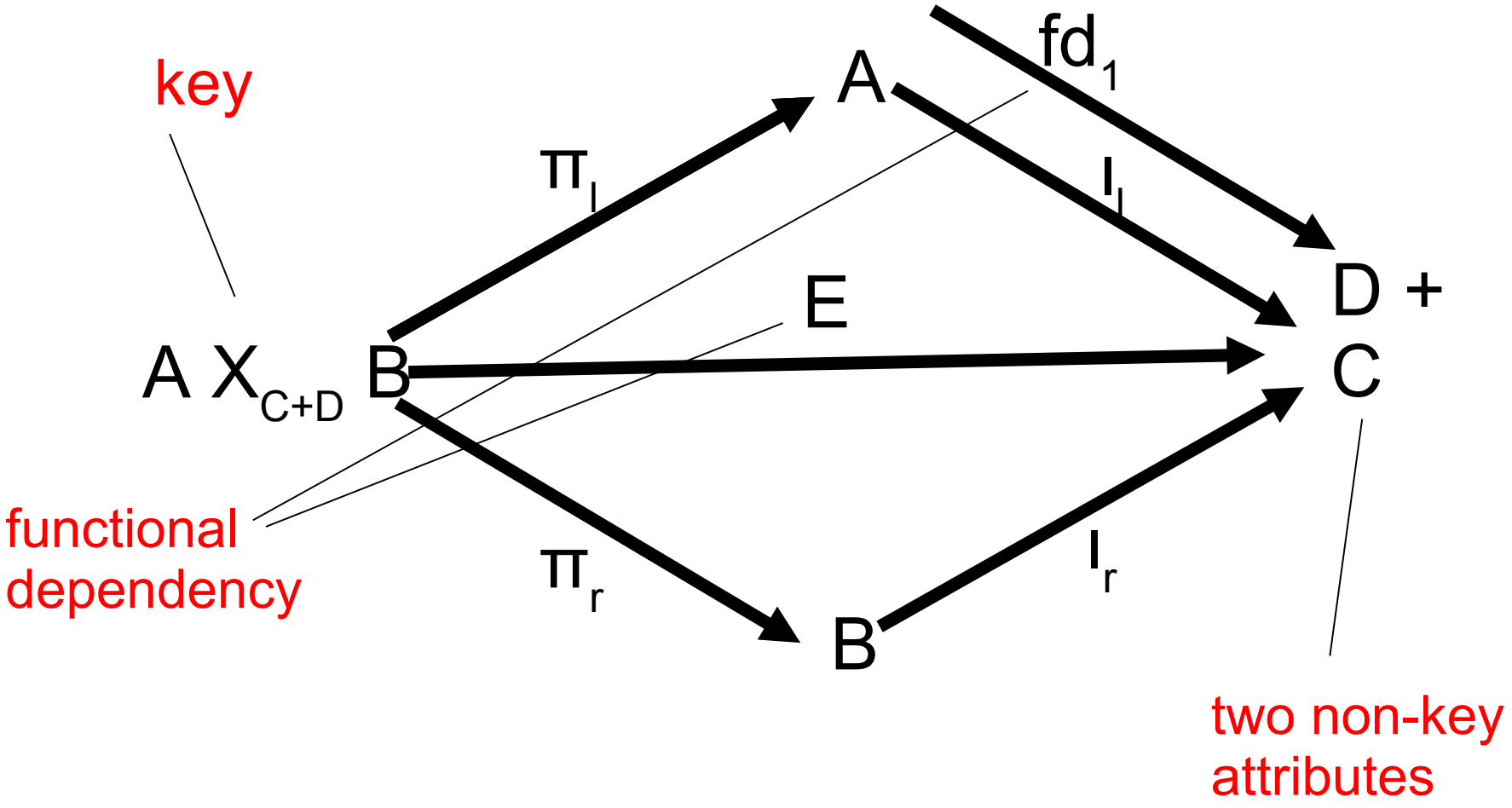


All attributes must be related;
adding stand-alone attributes means it is not even CCC

1NF is insufficient

- Everything is connected
- But may not be connected optimally
    - May be other arrows
        - From key component to non-key as a functional dependency
        - From non-key to non-key as a functional dependency
- Tests for these arrows are done in 2NF and 3NF respectively
- Potential presence of these unwanted arrows means that the diagram is not yet a LCCC

Introducing arrow to invalidate 2NF

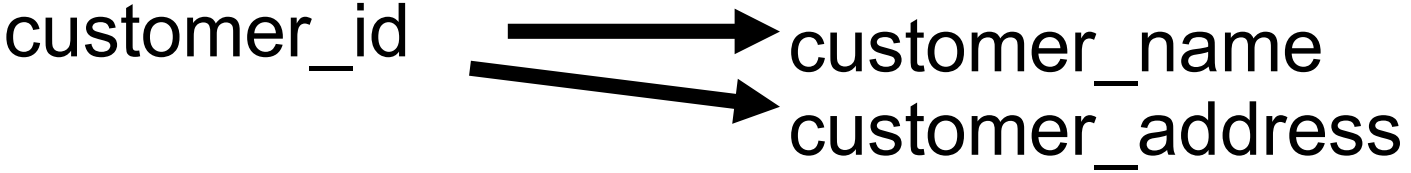


$fd_1 : A \rightarrow D$; $I_l : A \rightarrow A + B + C + D$;

adding fd_1 means that component of key determines non-key

Example of failing 2NF relation

Functional dependencies below are from component of key to non-key



Vast duplication of customer data each time something is delivered

Not a Valid Category, let alone LCCC

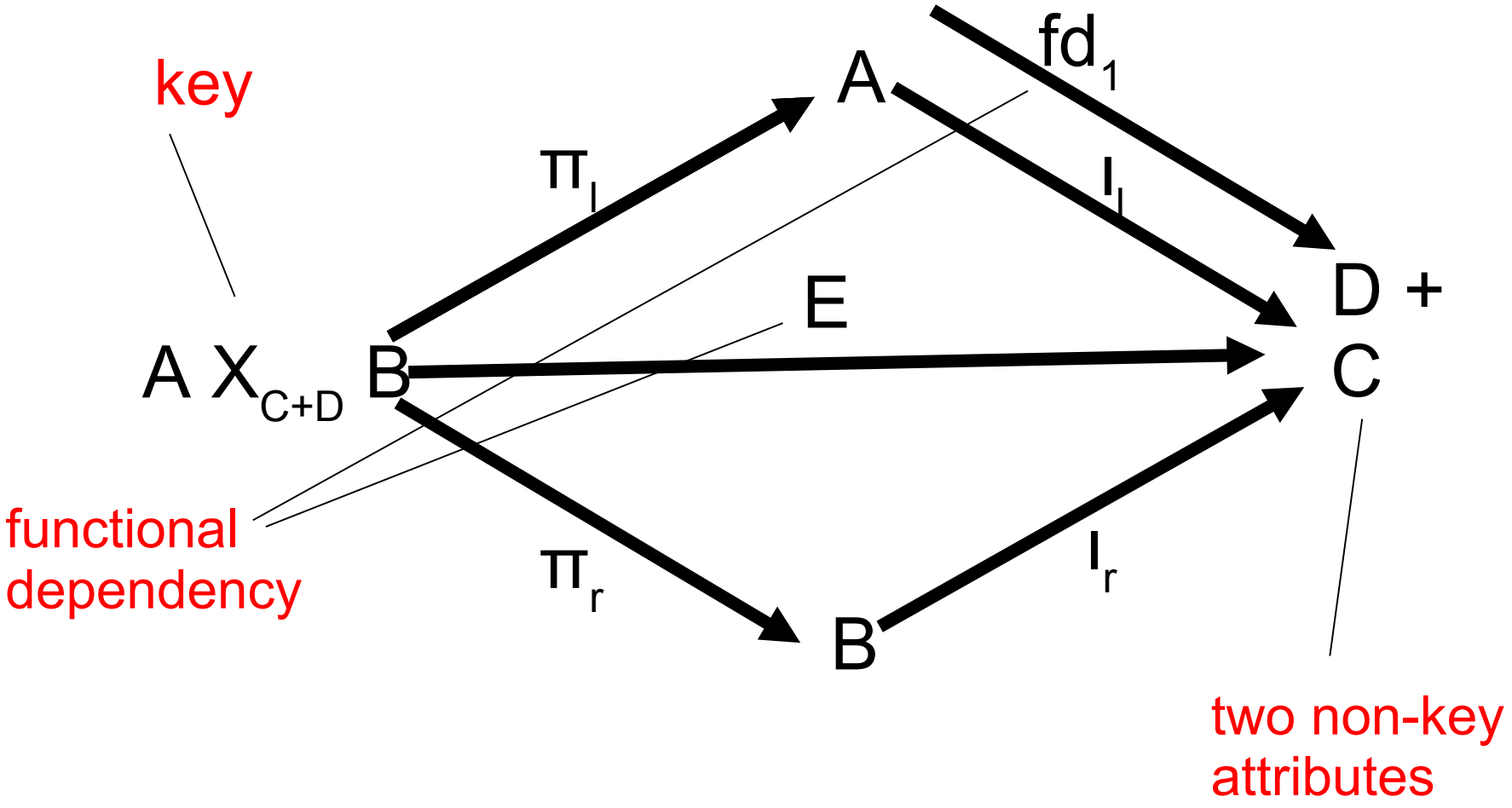
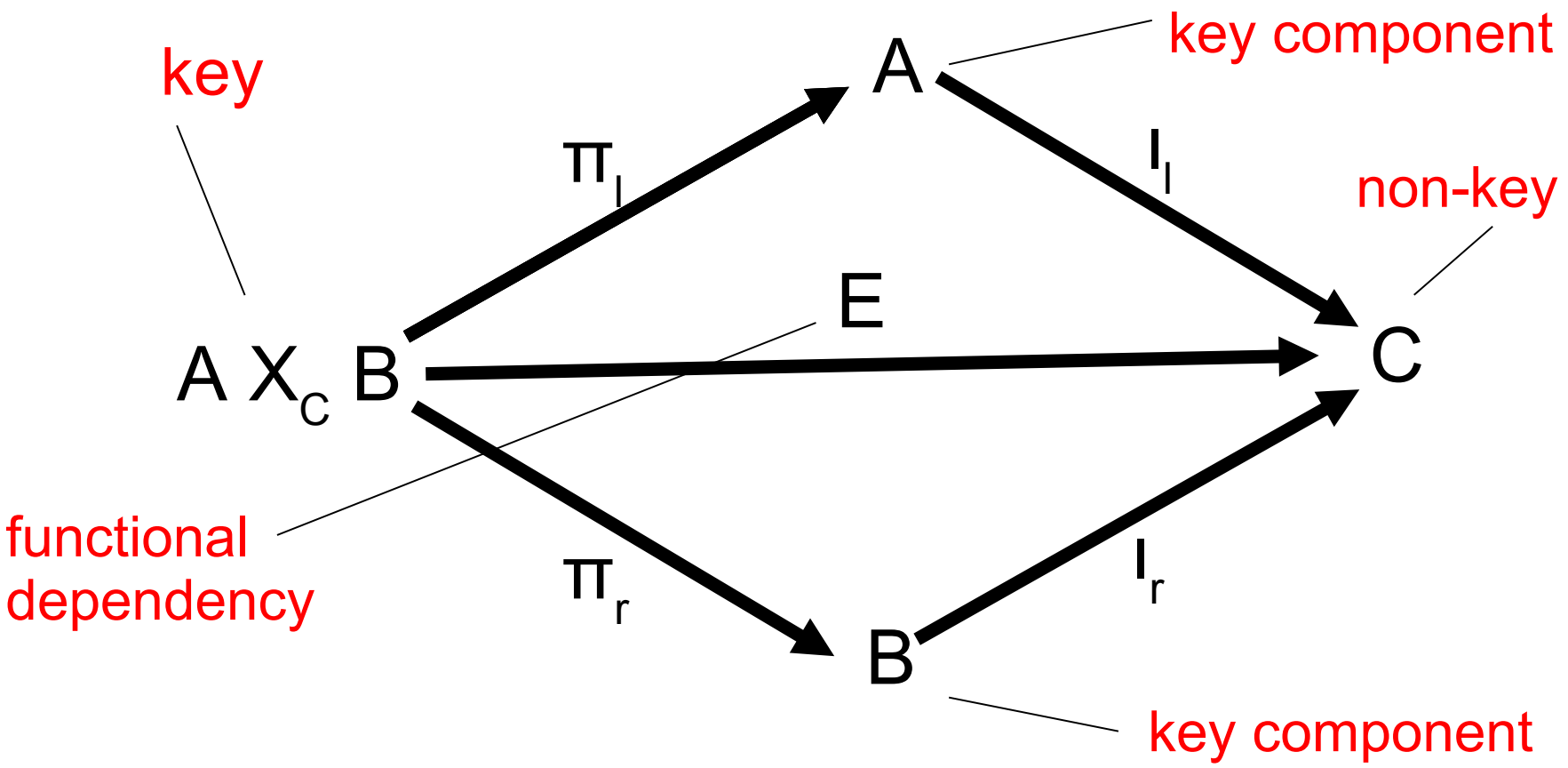


Diagram does not commute. D+C obtained by following top path does not equal that obtained by following bottom path.

Solution

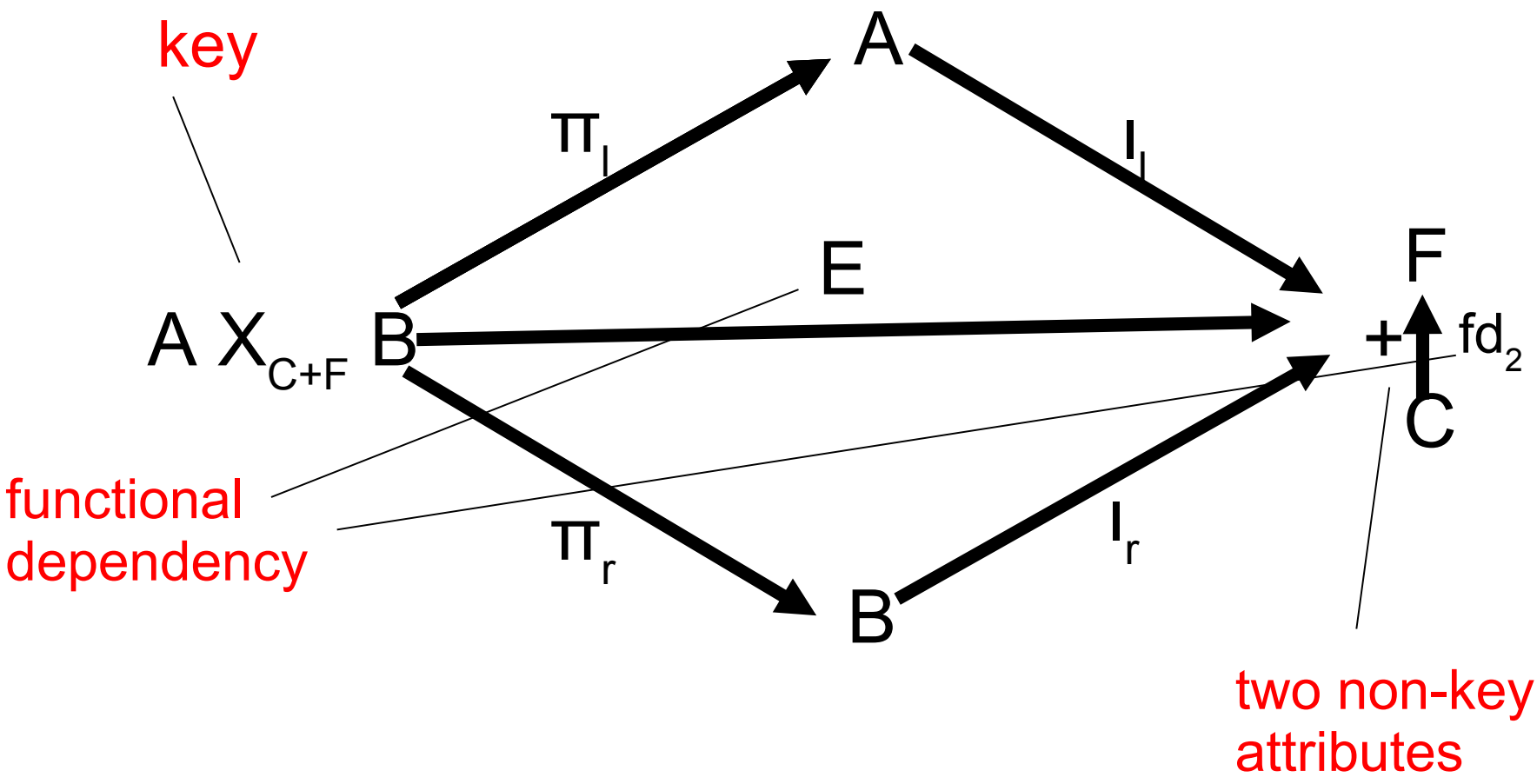
- Take A → D arrow out of pullback diagram
- Insert A → D dependency within category A, giving A more internal structure
- Alternative: possibly paste an additional pullback onto previous structure.

LCCC view of 2NF - Pullback



Category A contains dependency $fd_1 : A \rightarrow D$

Introducing arrow to invalidate 3NF



$fd_2 : C \rightarrow F$;

adding fd_2 means that one non-key determines another non-key

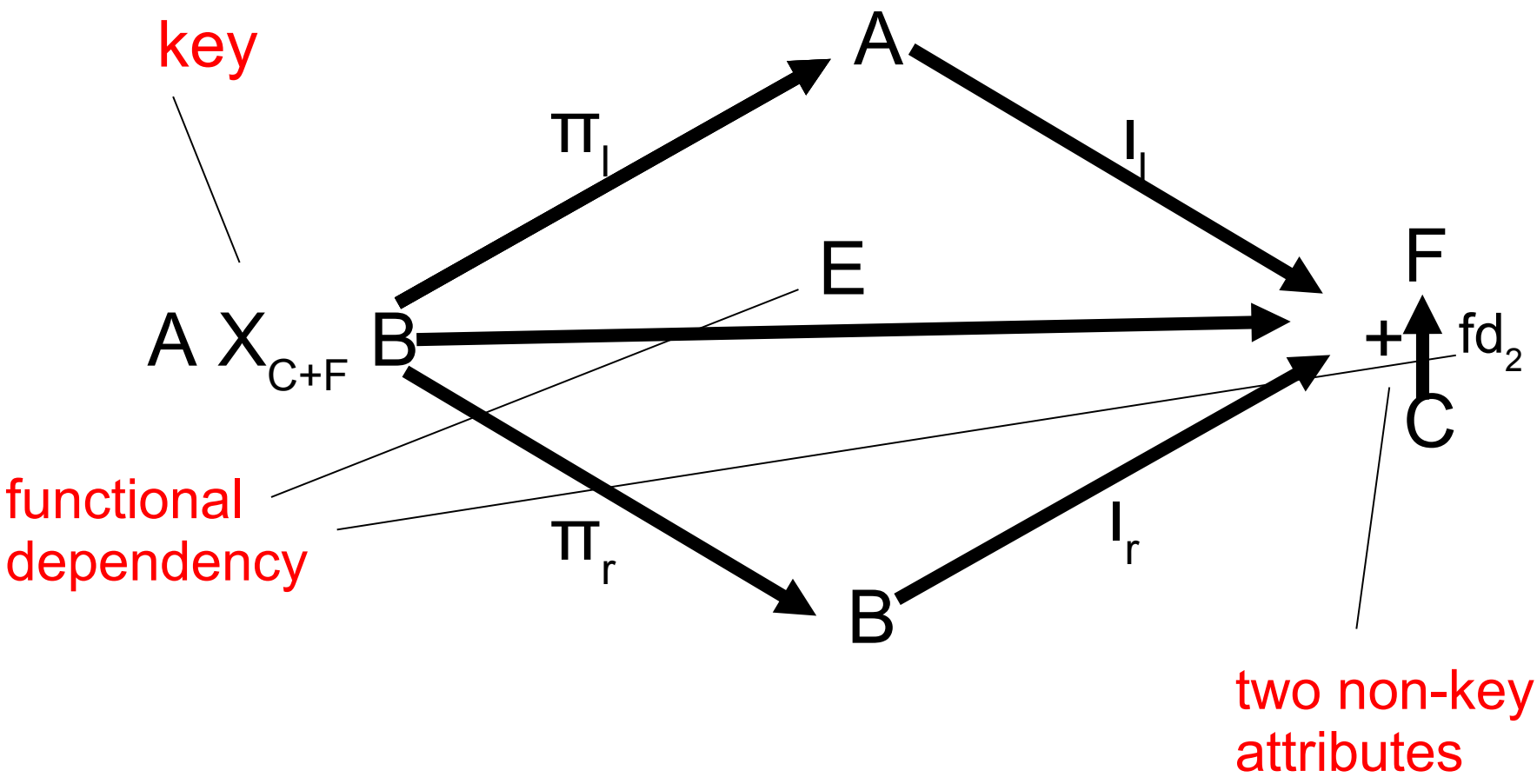
Example of failing 3NF relation

Functional dependencies below are from non-key to non-key

driver_id  driver_name

Vast duplication of driver data each time something is delivered

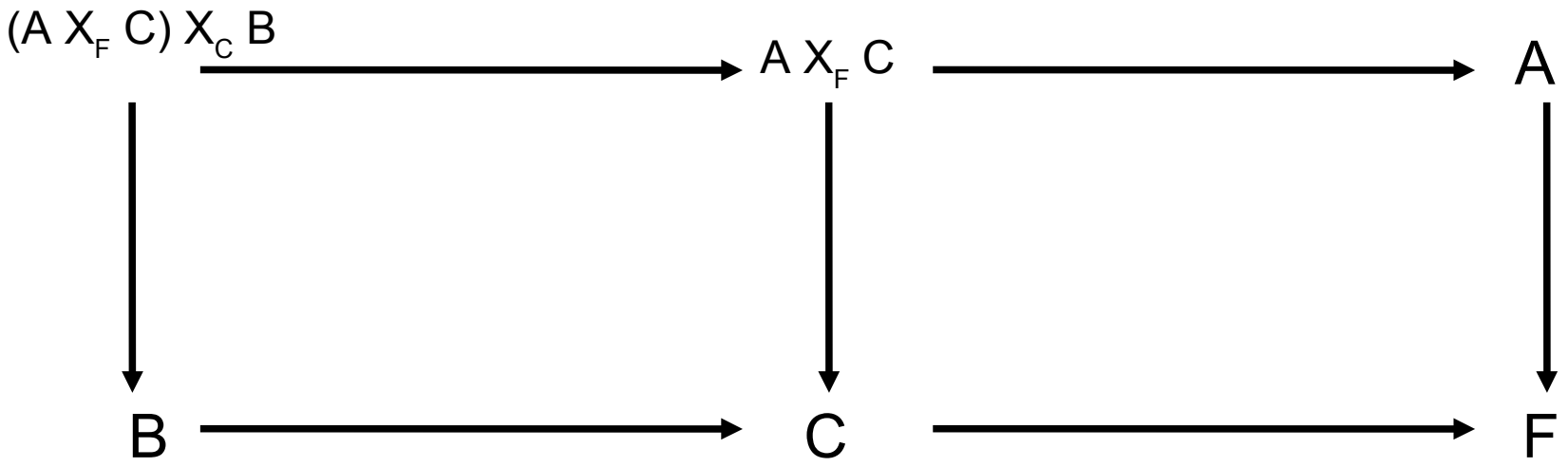
Not a Valid LCCC (Pullback)



Terminal object should be A+B+C+F (typed as a sum);
May not even be a category (depends on how constructed)

Solution

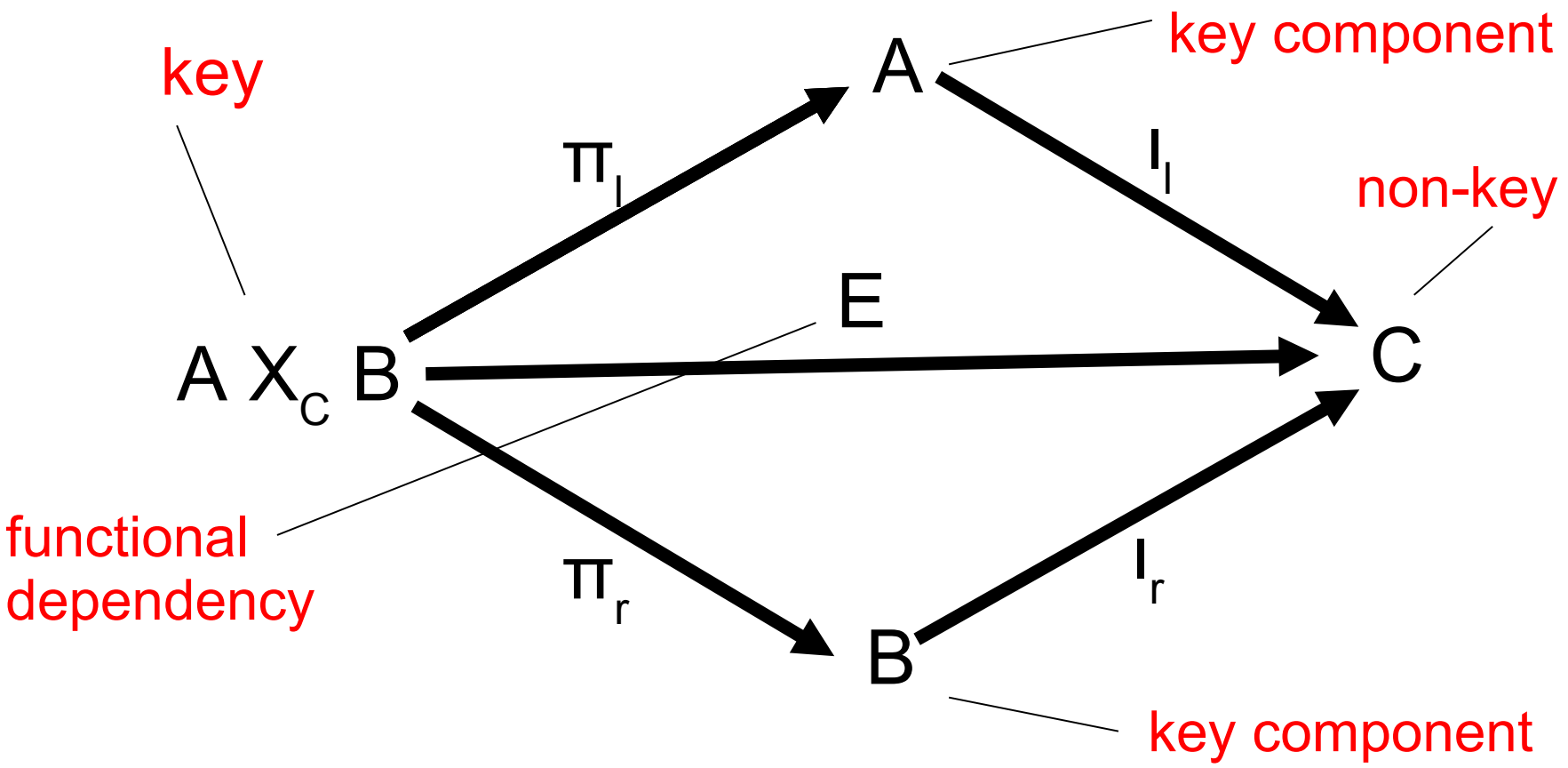
- Take C → F arrow out of pullback diagram
- Develop new pullback to represent relationship between C and F
- Paste new pullback onto existing structure.



3NF and LCCC

- 3NF (non-stepping stone via 1NF and 2NF)
  - A relation is in 3NF if each non-key attribute is dependent on the key, the whole key and nothing but the key
- LCCC
  - Relations are in 3NF if representable as a pullback

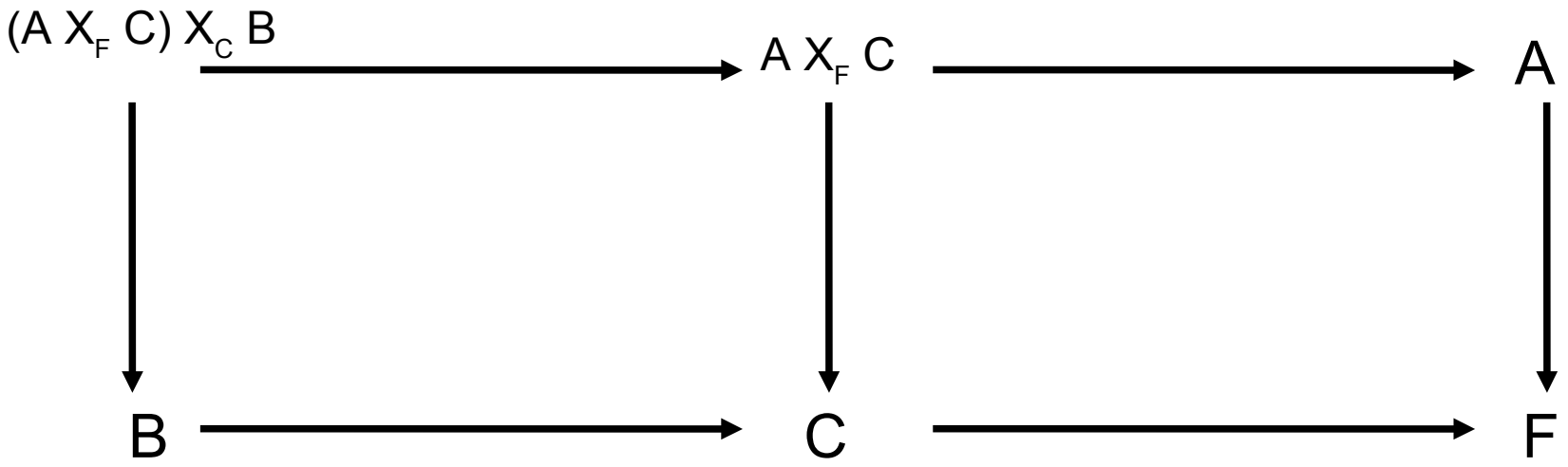
LCCC view of 3NF – Single Pullback Diagram



No other arrows permitted

LCCC view of 3NF – Pasted Pullback Diagram

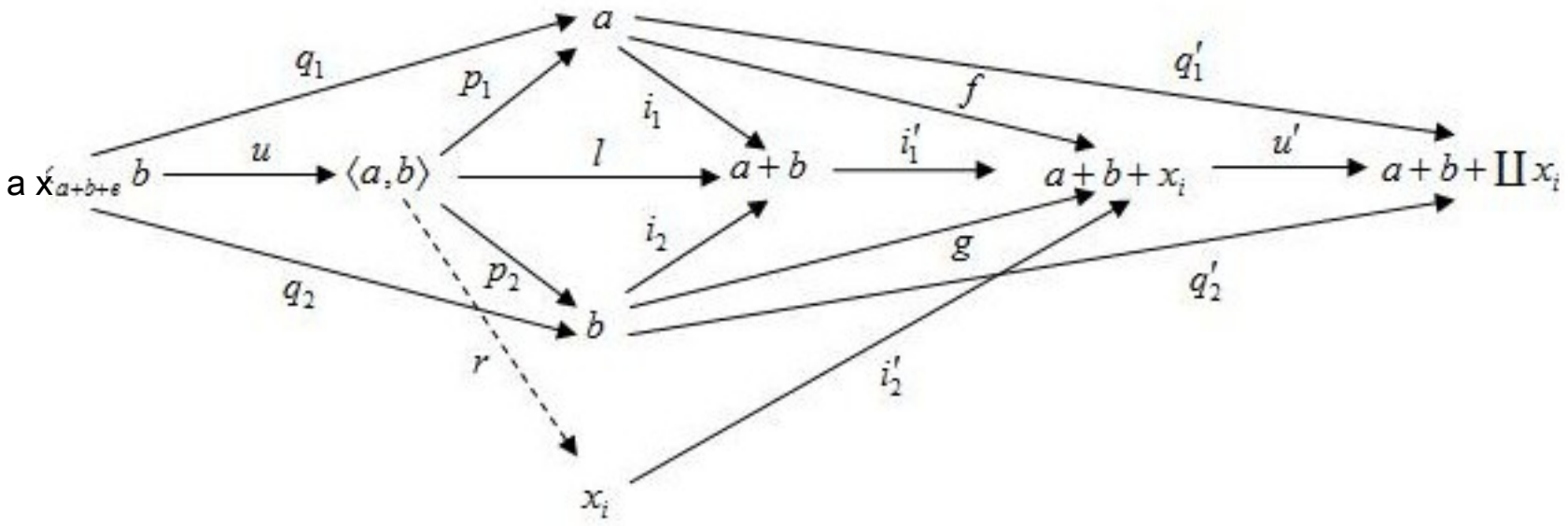
Complex pullback diagrams can be pasted together as below
Format of squares as below must be respected
No other arrows allowed



Higher Normal Forms

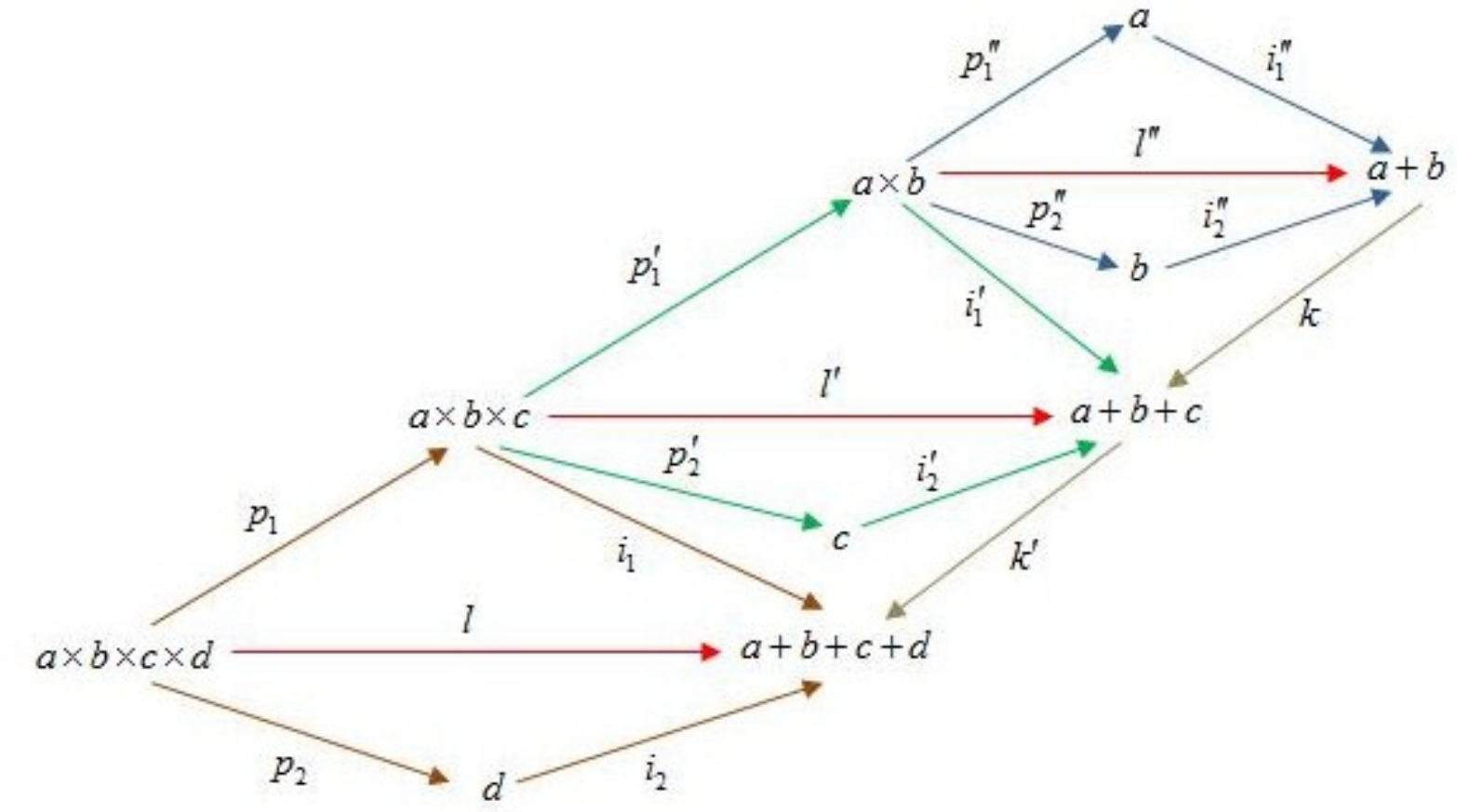
- In database theory go up to Boyce-Codd, 4NF and 5NF.
    - But 3NF is industry standard
- 5NF is Project-Join Normal Form
    - Define relations so that projection of attributes followed by joining together again returns starting point
- Already provided by LCCC in the adjointness between the X side and the + side.

Pullback showing both Intension and Extension



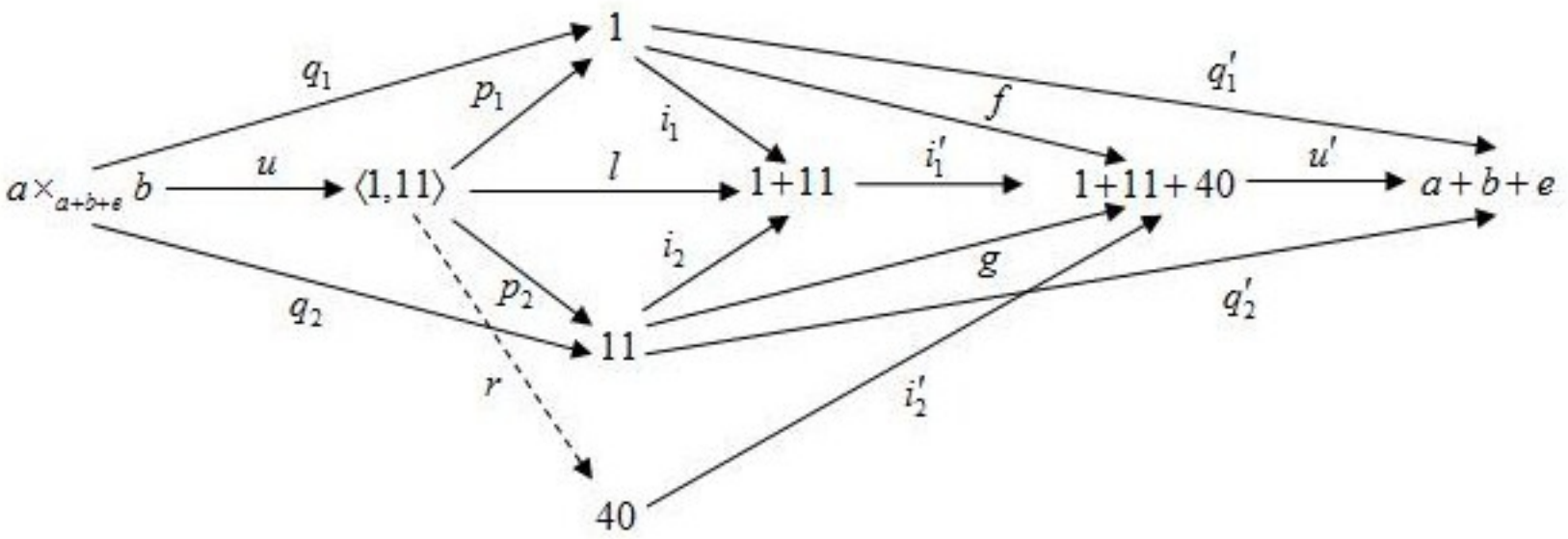
General Case for Single Relation in 3NF

Pullback showing both Intension and Extension



General Case for Pasting Relation with Composite Key in 3NF

Pullback showing both Intension and Extension

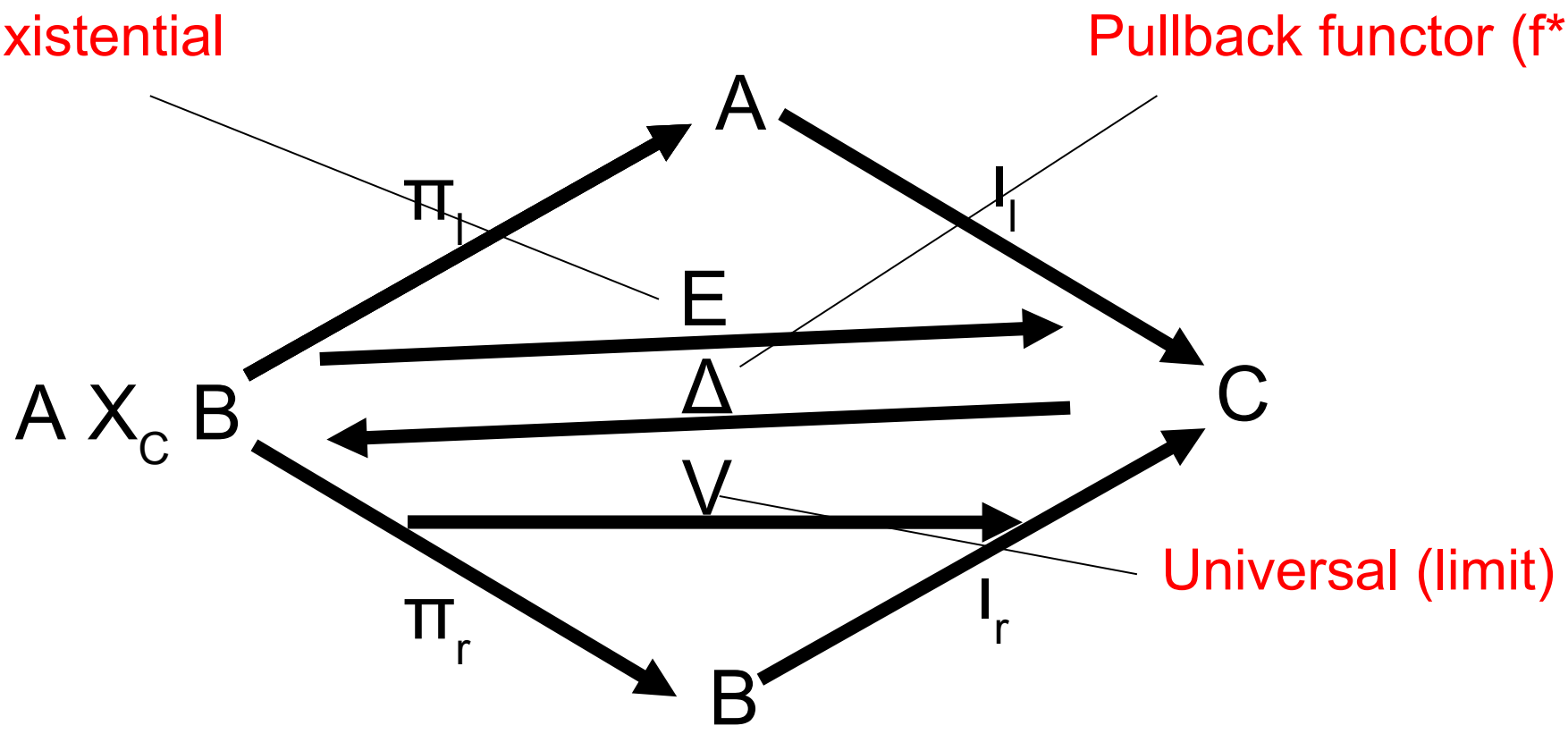


Single Relation in 3NF showing data values for extension

Higher Normal Forms

- In database theory go up to Boyce-Codd, 4NF and 5NF.
- But 3NF is industry standard
- 5NF is Project-Join Normal Form
- Define relations so that projection of attributes followed by joining together again returns starting point
- Already provided by LCCC in the adjointness between the X side and the + side.

LCCC for 5NF



Adjointness E ⊣ Δ and Δ ⊣ V between functors mapping between X and + (project-join)

Interesting Points

- So assumption that LCCC is a satisfactory basis for information system representation is justified by its close correspondence to data normalisation at industry standard (and beyond)
- Data normalisation has a sounder basis in LCCC than in set theory
    - Conceptual bases conform naturally
    - All normal forms up to 5NF are handled in a single diagram

Concluding Remarks

- LCCC are indeed justified as the choice of category for representing information systems
  - Data structures as pullback
  - Data normalisation to 3NF industry standard and beyond to 5NF
- Advantages of LCCC over Sets
  - 3NF is achieved directly through the pullback construction
  - Not through an optional external design process of normalisation, unenforced in relational database systems

Weak versus Strong Anticipation for Normalisation

- Weak anticipatory approaches include
  - Set-theoretic approaches such as Ullman
  - Industry approaches such as 1NF-3NF
  - Categorification approaches such as Rosebrugh
- As they are:
  - Models of the real-world
  - External to the operational system
- 5NF approaches LCCC in intent and purpose. It is still weak as external to system.

Weak versus Strong Anticipation for Normalisation

- Strong anticipatory approaches include
  - Locally Cartesian Closed Categories (LCCC)

- As they are:
  - Not a model (artefact) of the real-world
    - First-class construction in category theory
  - Inherently within the operational system
    - Enforced at all times through adjointness