

CG compared to OO networks

- *Components.* OO & CG: concepts and (conceptual) relations
 - implemented directly
- *Instances.* CGs: concrete, OO: all
 - method 'instances'
- *Hierarchies.*

OO: Inheritance hierarchies for object classes

CGs: Concept and relation type labels: sub (super) types

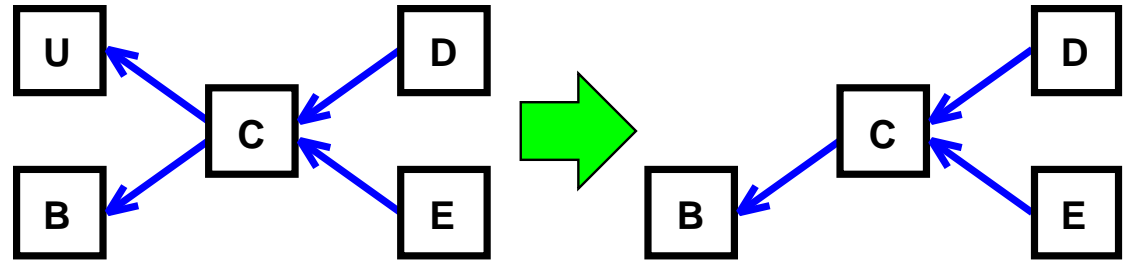
- instances of a meta-class 'concept-type'/'relation-type'
- attribute 'type' referring to the appropriate type class

Generalisation and specialisation of CGs

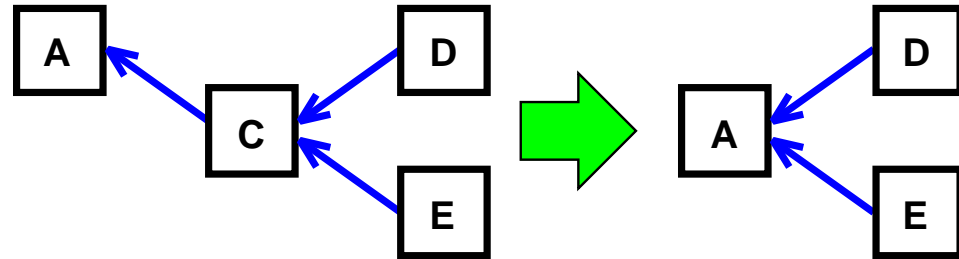
- special objects for representing CGs,
- methods for modification and combination.

- *Copy*. An exact copy may be made of any DG U .

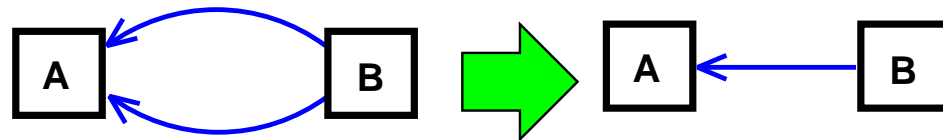
- *Remove unnecessary result concept U* :



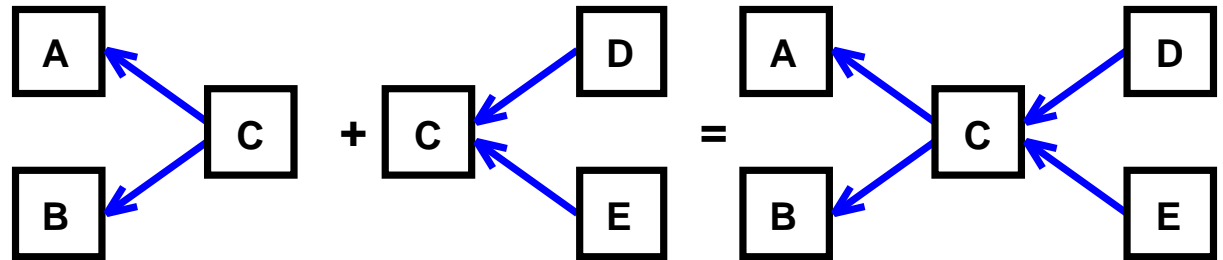
- *Remove intermediate concept C* :



- *Simplify duplicate relations*:



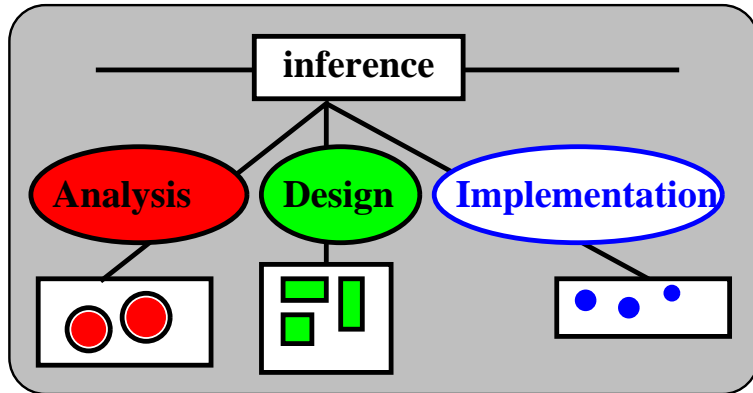
- *Join compatible concepts*:
 - equality
 - generalization: domain-dependent
 - hypothesis combining knowledge



Inference model: Integration of three models

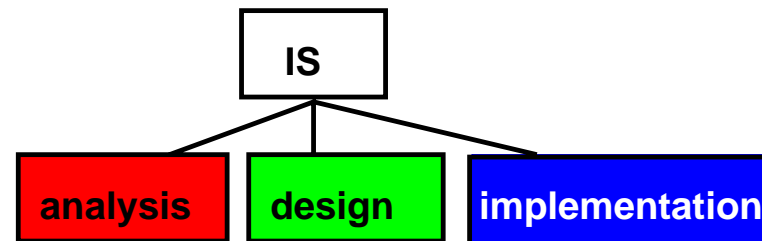
Inferences

- three attached descriptions
- different amounts of items



Shared inference structure (IS)

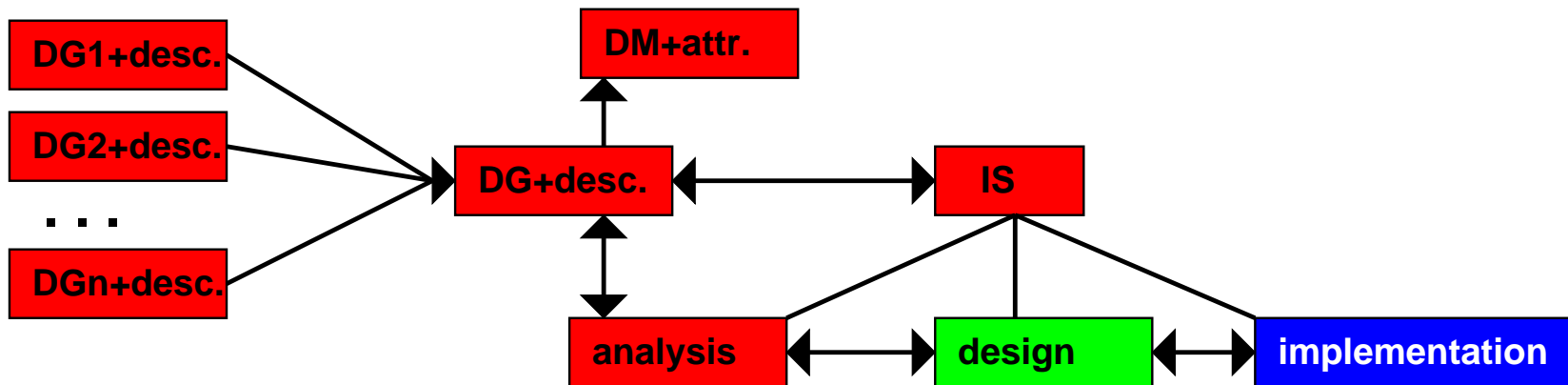
- three sets of descriptions
- traceability



KB construction: details in analysis (some iteration),
formalized in design and implementation

Seamless transformations and development

- templates for semi-automatic transformations
- mostly two-directional
- traceability, careful utilization of work already done



Seamless development: sequence of seamless transformations

- Seamless Structured Knowledge Acquisition (SeSKA) – **abstract**
- Structured Object-Oriented Knowledge Acquisition (SOOKA) – **OO**

Structured representation

Attributes in roles, dependency graph, shared inference structure

