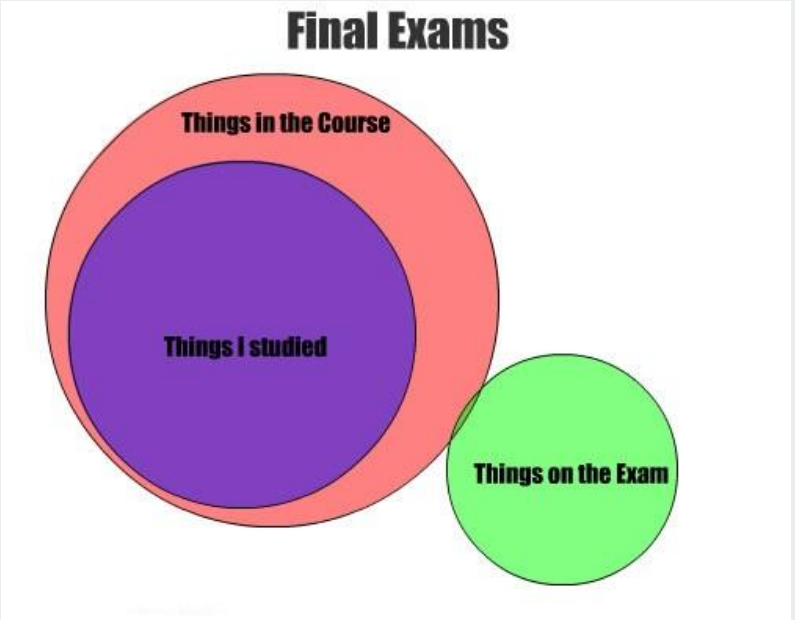
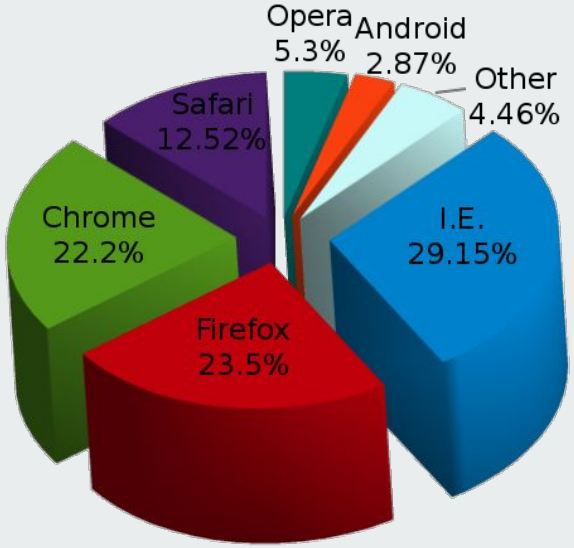
On Visual Formalisms

Paper Published: May 1998
Author: David Harel
Presenter: Mike Maksimov



Introduction to Visual Formalisms



Browser Usage on Wikimedia
December 2011

Why do we need visuals (e.g. graphs)?

- We live in a visually driven society.
- A visual image simplifies interpretation of data.
- Big amounts of data can be represented in a smaller form, making assimilation easier.

Graphs in Computer Science:

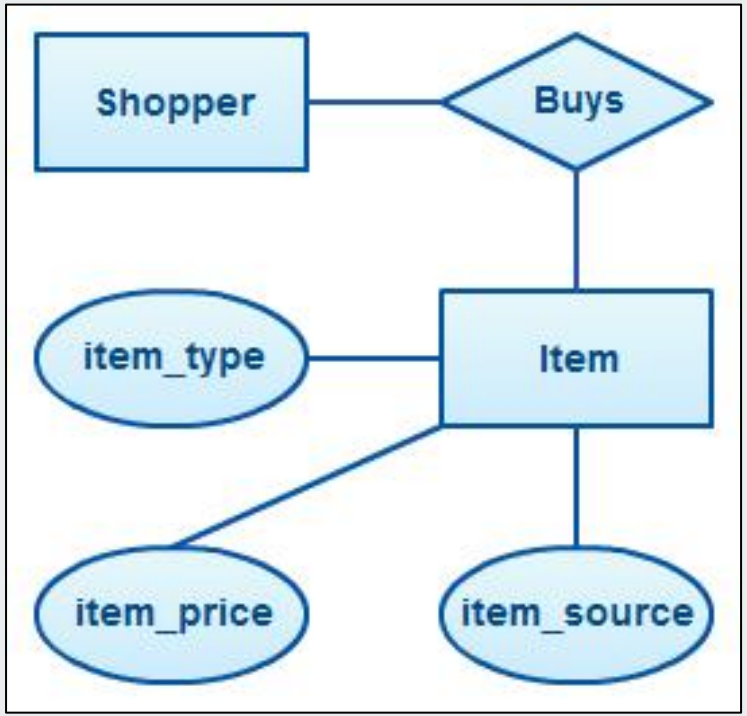
- Logical Circuits
- Activity Diagrams
- State Diagrams
- Entity Relationship Diagrams
- Etc.

First we will look at:

The work of Leonhard Euler. Creator of the two well known topo-visual formalisms.

- The Formalism of Graphs
- Euler Circles

Types of topo-visual formalisms



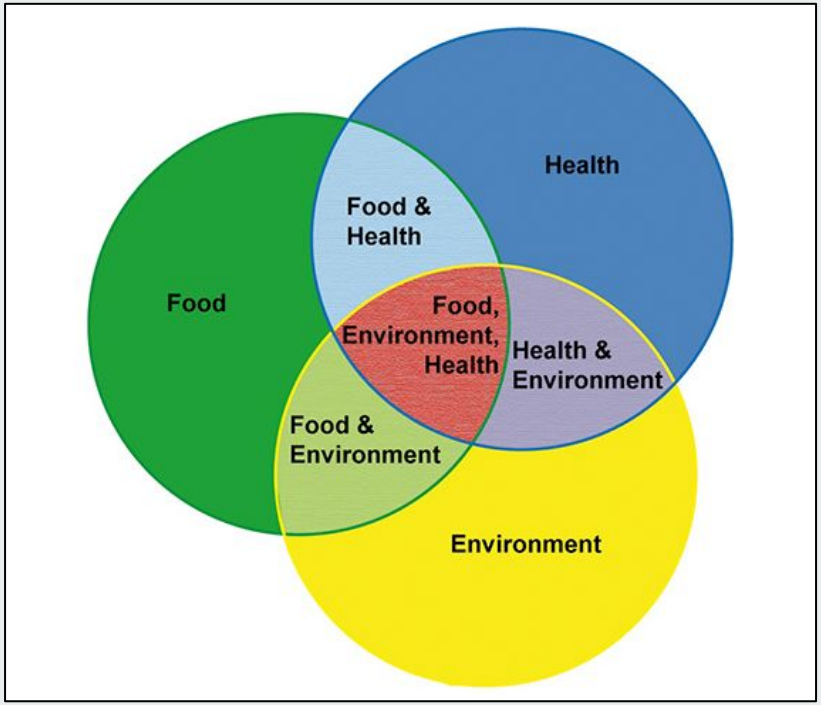
Definition of a graph in its most basic form:

- A set of points, or nodes, connected by edges or arcs. The role is to represent a (single) set of elements ***S*** and some binary relation ***R*** on them.
- The precise meaning of the relationship ***R*** is part of the specific application and use case, and can represent any kind of relationship.
- The nodes similarly can represent a range of the most concrete to the most abstract examples.

Good for:

- Representing a set of elements together with a special type of relation(s) on them.

Types of topo-visual formalisms



Euler Circles/Venn Diagrams

- Closed curves partition the plane into disjoint inside and outside regions.
- A set is represented by the inside of a curve.
- This gives the topological notions of ***enclosure***, ***exclusion*** and ***intersection***.

Good for:

- Representing a collection of sets, together with some structural (i.e. set-theoretical) relationships between them.

There is a problem!

Observation

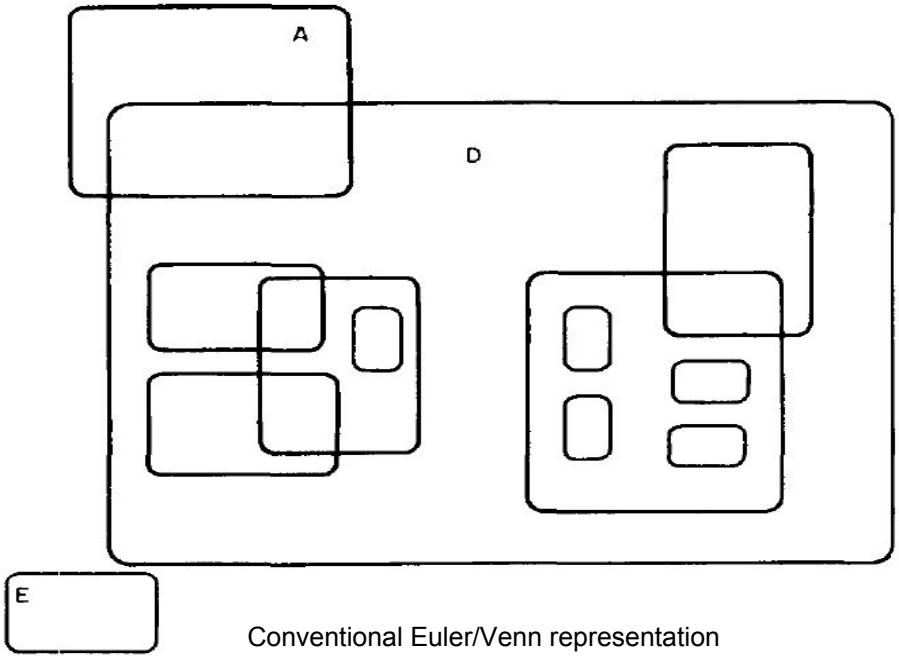
- In many cases, both capabilities are needed.
- In order to compensate for this fact, many of the visualizations are far more complex than they have to be.
- It is often also desirable to identify the C**artesian product** of some of the sets. Something that the previous formalisms lacked.

Solution = Higraphs

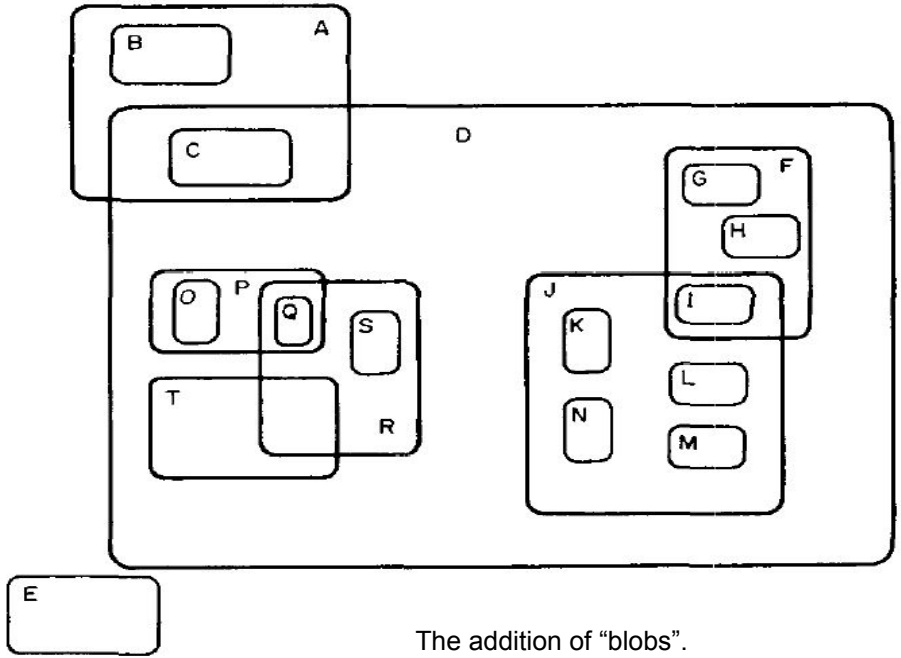
Higraphs

Characteristics:

- Higraphs modify and combine Euler's two topo-visual formalisms into one, supporting the capabilities of both.
- They are also extended in order to easily represent Cartesian products.
- Higraphs are ultimately Euler circle curves connected to each other by edges or hyperedges.



Conventional Euler/Venn representation

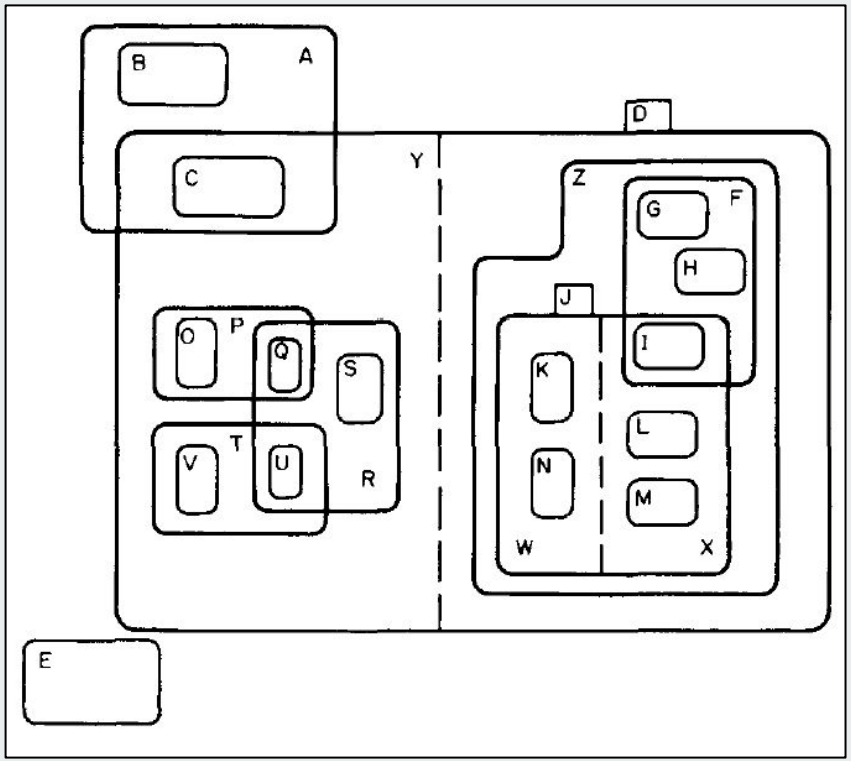


The addition of “blobs”.

- Every set is labeled by a “blob” for easier reference.
- Blobs that hold no other blobs are called “atomic” sets.

- If there is an intersection without a blob inside (e.g. ***T*** & ***R***), that intersection does not mean anything.
- Explicit blobs allow the reference to the difference of blobs (e.g. A - D).

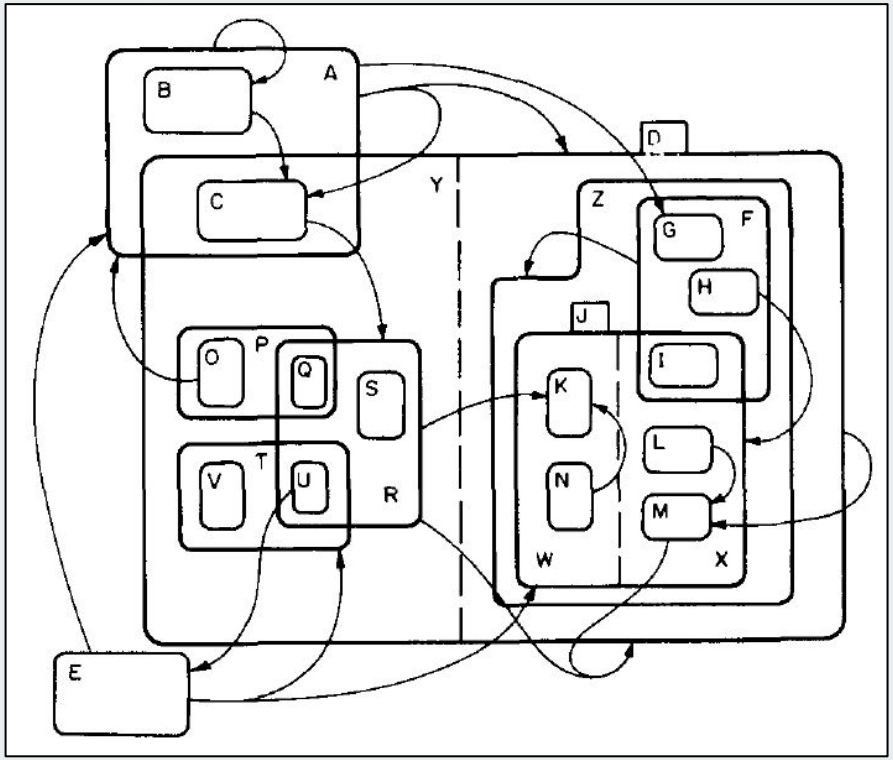
Adding Cartesian Products



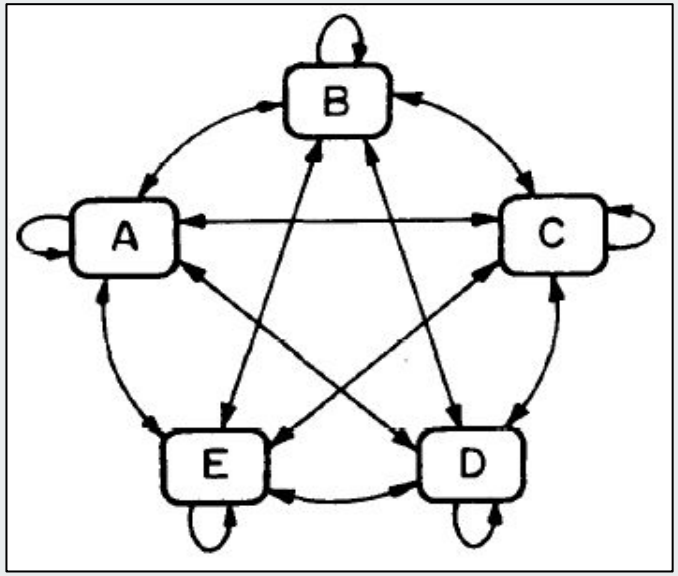
- The notation is a partitioning by dashed lines.
- **J** is no longer the union of K, N, I, L, M. It is now used to represent the product of the union of elements.

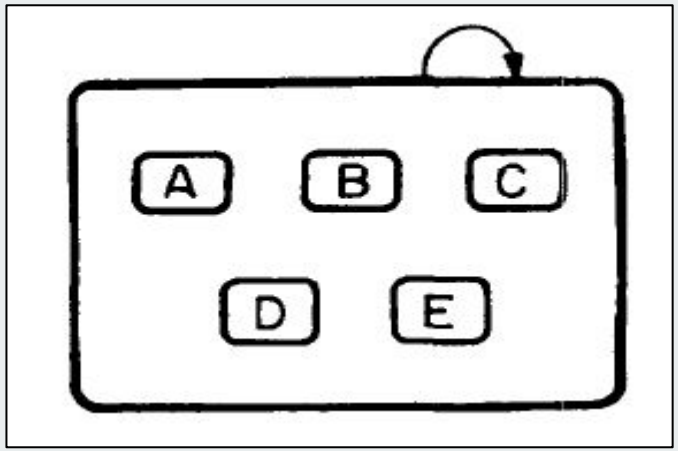
  J = W **x** X = (K&N) **x** (I&L&M).

Adding Edges



- Edges are sorted into high-level (e.g. **E** to **A**), low-level (e.g. **N** to **K**) and inter-level (e.g. **U** to **E**).
- Each set of interest having its own contour also enables more connection flexibility.





Higraph Applications

Use cases:

- E-R diagrams
- Activity charts
- State diagrams
- Etc.

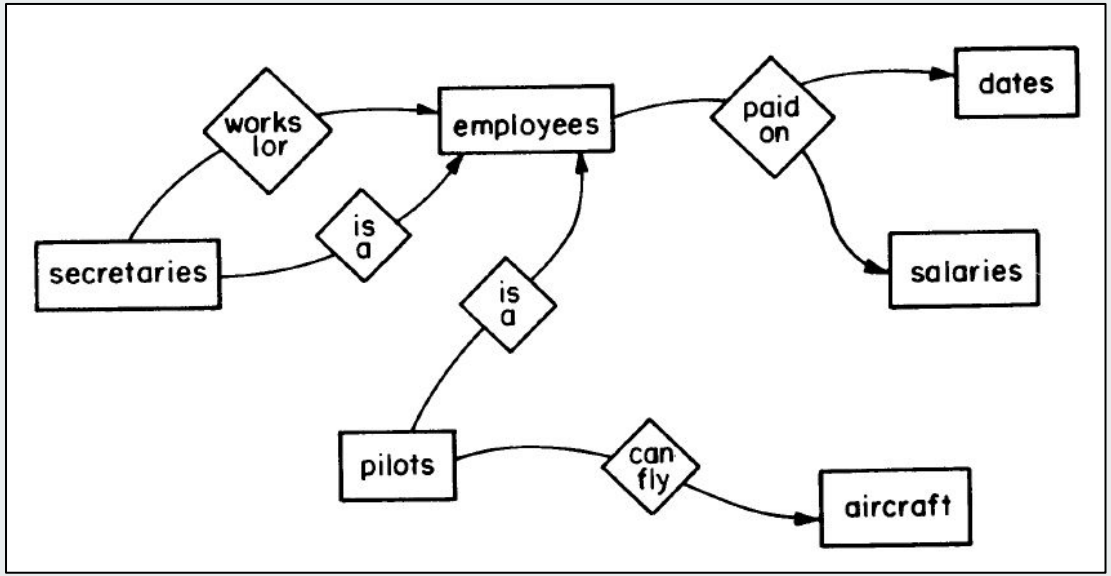
What benefit do higraphs add to these cases?

- Higraph edges are not limited to connecting elements to elements, but can connect **sets** to **sets**.
- They are not “flat”, and support a notion of hierarchy.
- Reduce clutter, resulting in a more clear and concise picture.

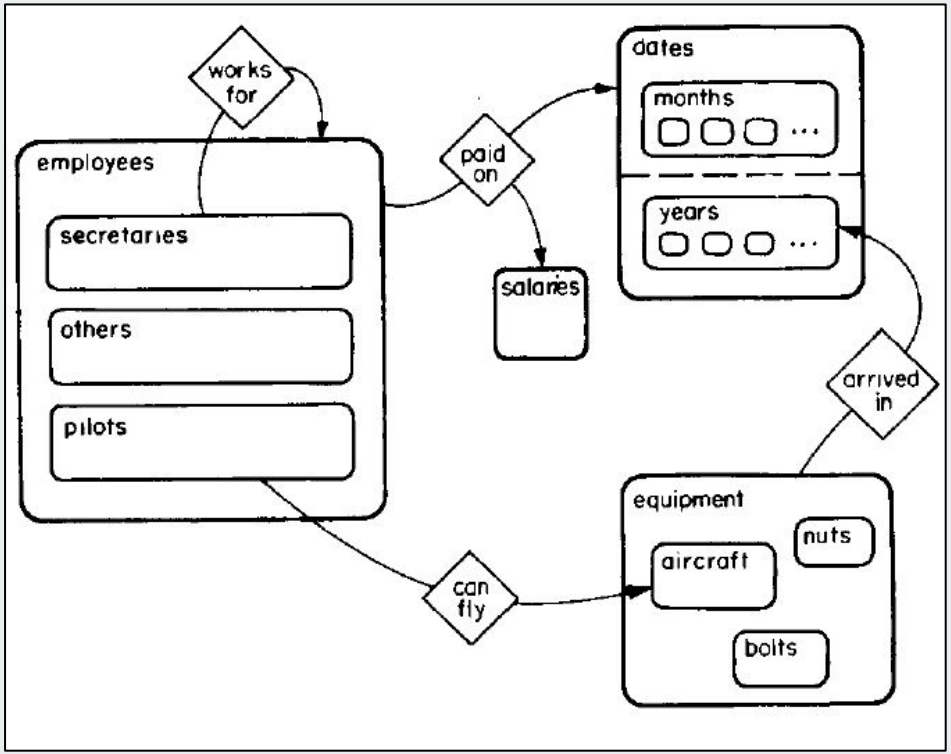
E-R diagram Standard Example

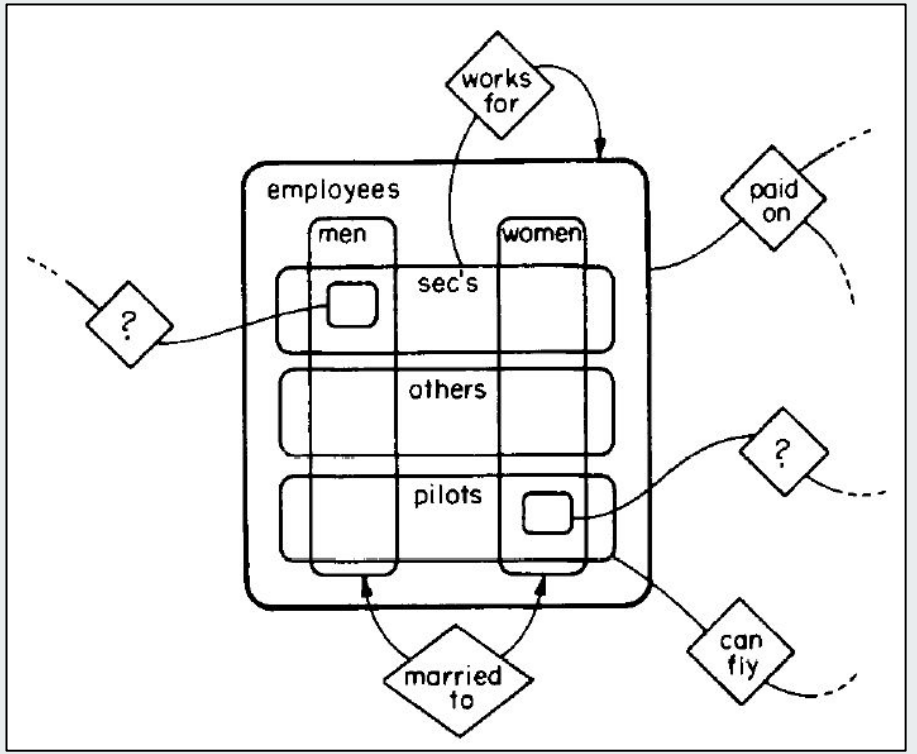
Cons:

- The diagram is “flat”.
- The “**is a**” nodes convey information of the structural, set-theoretical relationship type.
- Explicitly having to place such nodes each time will cause clutter and become unmanageable in more complex diagrams.

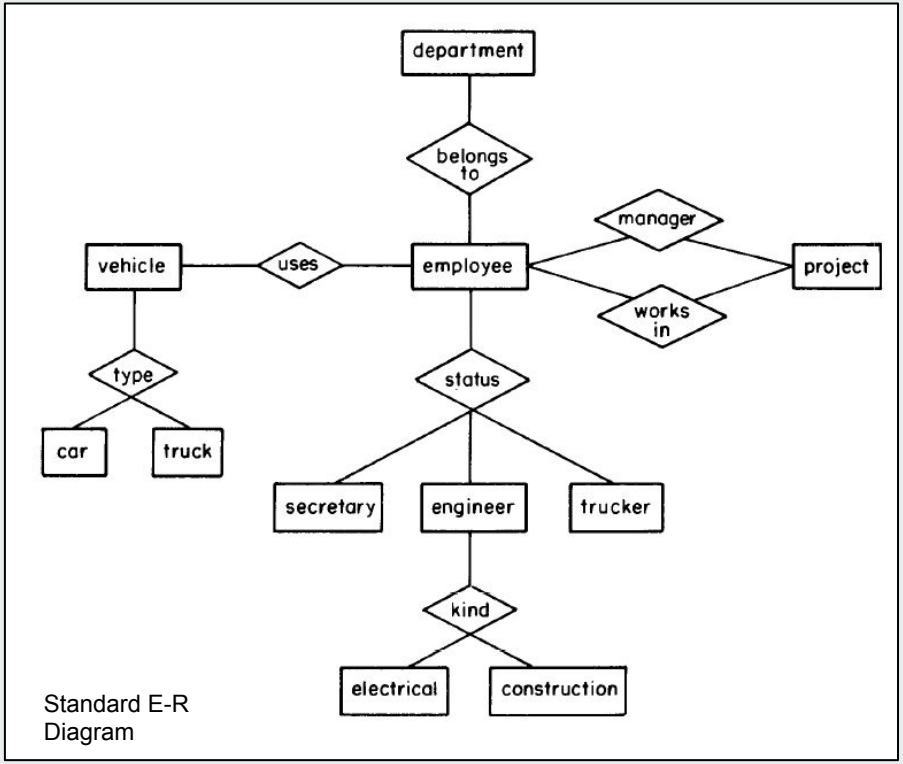


E-R diagram - Higraph Implementation

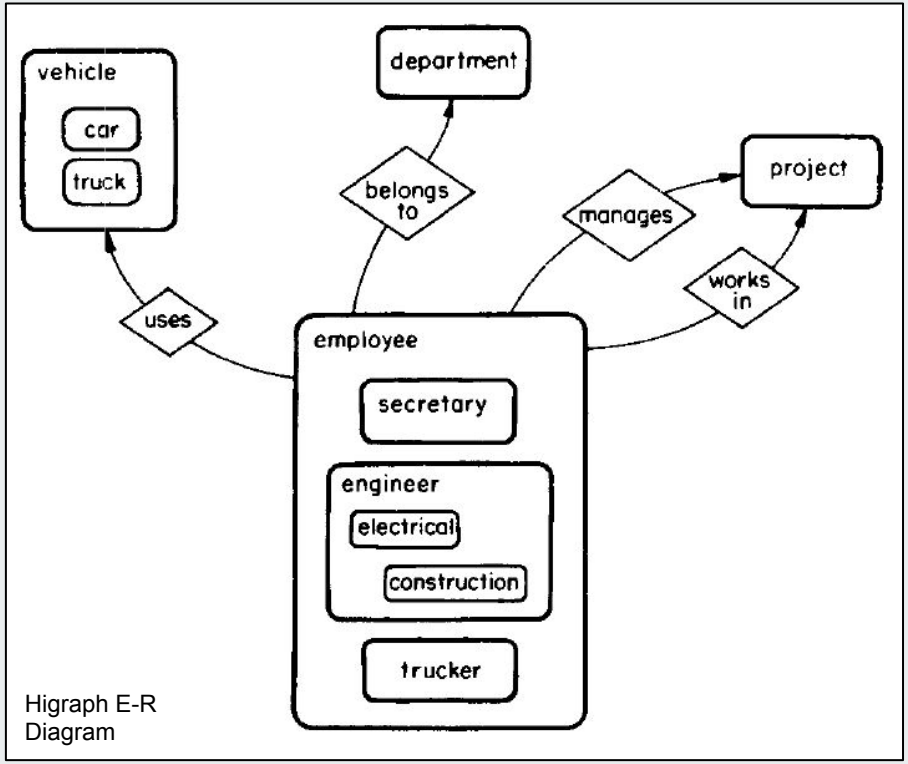




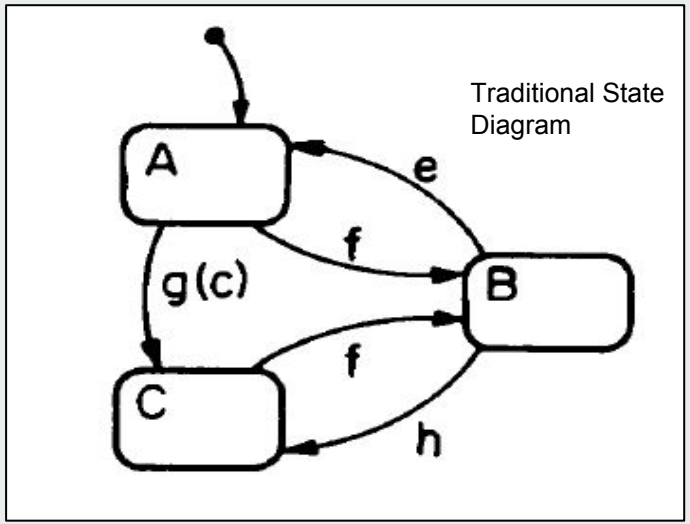
E-R diagram Example #2

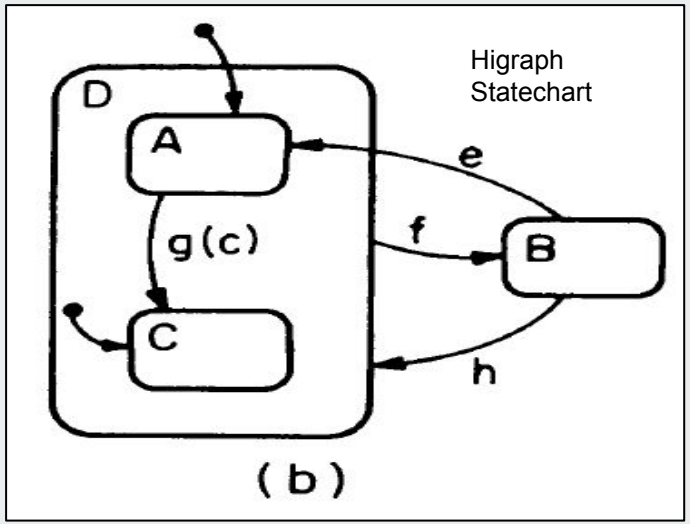


Standard E-R Diagram



Higraph E-R Diagram





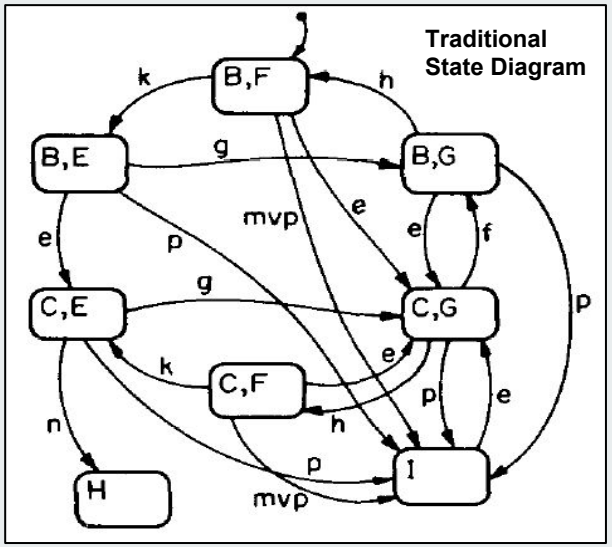
State diagram example

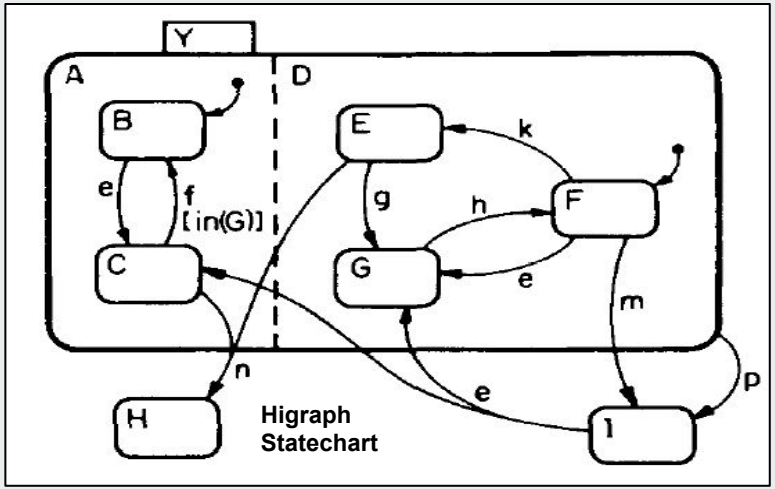
State diagram cons:

1. They are “flat” and provide no natural notion of depth, hierarchy or modularity.
2. They are uneconomical with regards to representing transitions.
3. Sequential in nature. They do not cater for concurrency in a natural way.

Statecharts - Higraph extension of state diagrams.

Statecharts = State Diagram + Depth + Orthogonality + Broadcast Communication.





State diagram example #2

Higraph Statechart benefits:

- They capture orthogonality by the partitioning feature of Higraphs, that is, by the unordered Cartesian product.
- The transitions are much easier to follow due to less clutter.
- There is a notion of depth/hierarchy.
- The problem of the exponential growth blowup is somewhat mitigated.

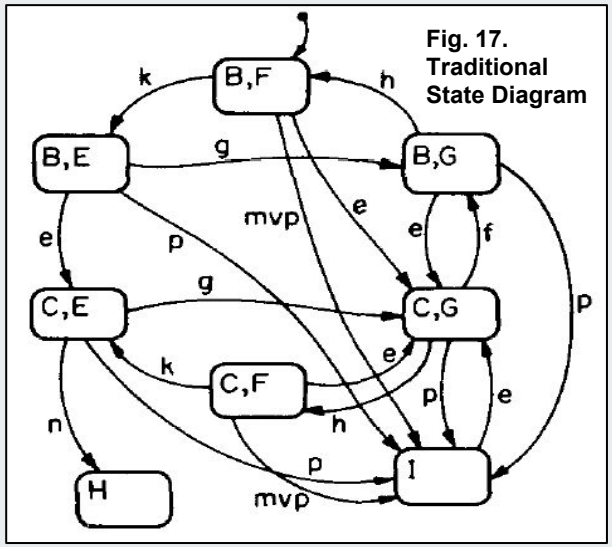


Fig. 17.
Traditional
State Diagram

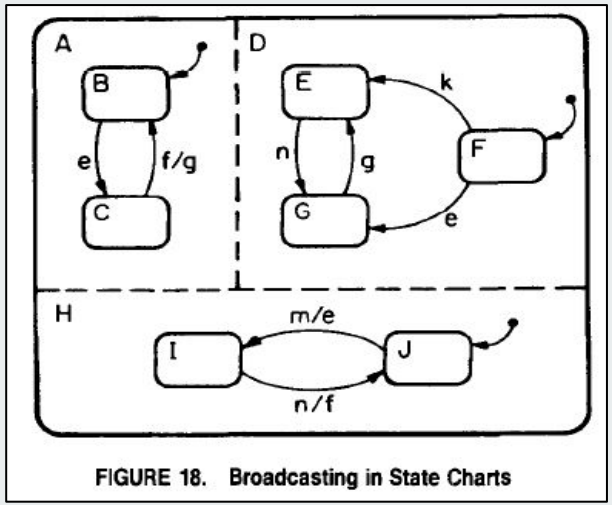


FIGURE 18. Broadcasting in State Charts

State diagram example #3

Benefit of broadcasting:

- Transitions are much easier to follow.
- **Example (fig. 18)**: If we are in (B, F, J) and an external event ***m*** comes in, then the next configuration will be (C, G, I), by virtue of ***e*** being generated in H and triggering the two transitions in A and D.

Last Note

A final lesson to take with you

- Graphical notations and formalisms are important for expressing information.
- They are extremely diverse.
- There is not one set that can be used for everything.

Discussion

Pros and Cons