Informatics
The Science of Minimal Systems with Maximal Complexity

Andreas Schwill
Institut für Informatik
Universität Potsdam
www.informatikdidaktik.de

Minimalism: my first unconscious contact:

A children's book entitled

A. Hageni (1965): "Onkel Puck mit der Posaune"

"Uncle Puck with the trombone"

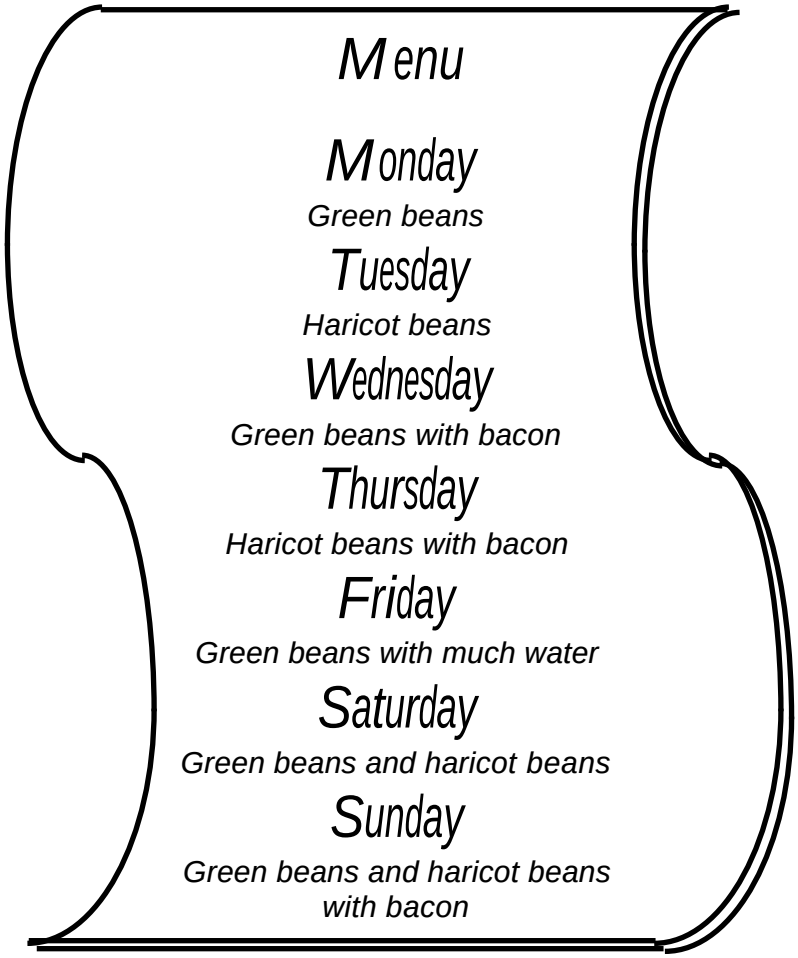
Uncle Puck is going on vacation with two of his nephews. They live in a small hut somewhere in a forest.

Uncle Puck a bit an odd bird, but a creative cook.

With only

green beans, haricot beans, and bacon

he prepares "wonderful" dinners for the kids according to the following menu:



After my lecture you would say:

with the construction kit

$$(\{G,H,B\},\{\cup\})$$

defining a generating system he constructs a set of dinners namely the closure

$$D=(\{G,H,B\},\{\cup\})^*$$

OVERVIEW

Overview

- **Introduction and basic notions**
- **Minimal systems in arts, science, and industry**
- **Orthogonalization as a fundamental idea of informatics**
- **Minimal systems in informatics**
  - **Machines**
  - **Control structures**
  - **Functional programming**
  - **Modelling**
- **Minimal systems in informatics lessons**
- **Conclusions**

1 Introduction and basic notions

complexity

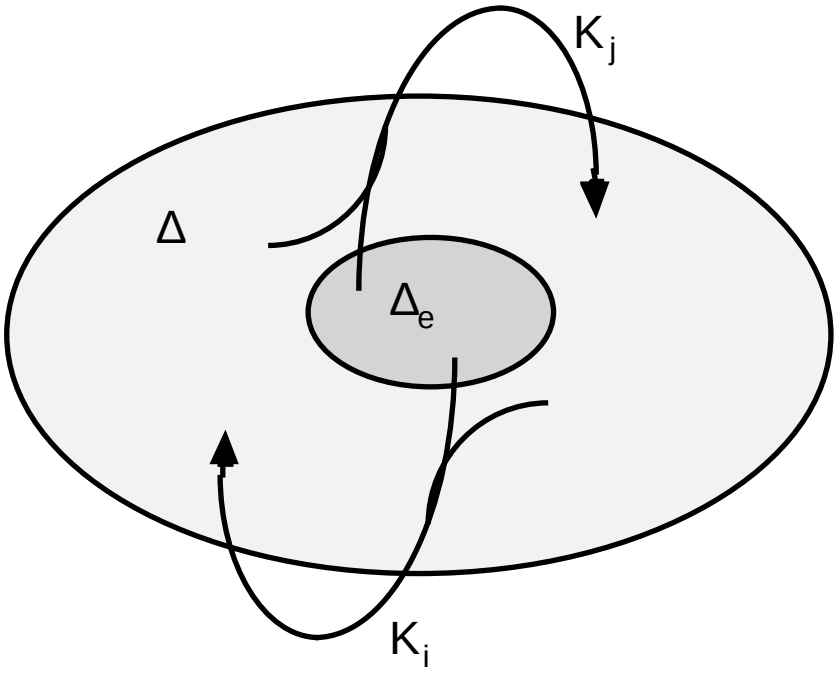
- **property of a system to be vast and diverse in its inner structure**

complication

- **property of the description of a system to be vast and varied and hard to grasp**

Minimal system – orthogonalization of a field Δ

- **small set Δ_e of elementary objects**
- **small set $K=\{K_1,...,K_n\}$ of simple operations on the basis**
- **minimal generating system $B=(\Delta_e,K)$**
- **construction kit for the field Δ**



2 Minimalism and orthogonalization in arts, sciences, and industry

Albert Einstein: **Everything should be made as simple as possible, but not simpler.**

David Hilbert (quoting an ancient French mathematician):
A mathematical theory cannot be considered perfect until it is elaborated so deeply that it can be explained to any guy on the street.

C.A.R. Hoare: **There are two ways of constructing a software design; one way is to make it so simple that there are obviously no deficiencies, and the other way is to make it so complicated that there are no obvious deficiencies. The first method is far more difficult.**

Why this human inclination to simplicity?

- **economically/ecologically: maximize results with minimal resources**

- **G.A. Miller (1956): The magical number seven plus or minus two**
  -> limited capacity of the short term memory
  -> most minimal systems contain less than 10 elements

Industry

platform strategy – lean production – lean management

51 models – 4 platforms (60% of the parts) – any combinations

<Picture of Volkswagen models>

wedding

<Picture of a wedding>

Minimal arts

- **elementary objects: colors, shapes, lines and textures**
- **creative formal combination -> vast number of artwork**
- **construction instead of representation**

<Picture>

Hoh Yin Ping - Ink traces

<Picture>

Yves Klein – Untitled Blue Monochrome

<Picture>

Piet Mondrian – Composition B with Red

Minimal music

- **elementary objects:**
    - **tones**
    - **rhythms**
    - **musical patterns**
    - **other compositorial means**
- **operations:**
    - **repetition of patterns**
    - **phase shifting**
    - **overlaying**
    - **stressing**
    - **adding single notes to change rhythms and sequences of tones**
- **simple in structure -> highly creative feeling of sound**
- **techno music -> hypnotic and ecstatic experiences**

Examples

- **Philip Glass - Mad Rush for Piano**
- **Antiloop – I love you**

Mathematics

Chaos theory: simple construction kits

Example: Julia sets

- **given f: $\mathbb{C} \rightarrow \mathbb{C}$ with $f(x)=x^2+c$ with constant c**
- **compute f^n**

<Picture of a Julia set>

Example: Kolmogorov complexity

- **formalizes the notions of complexity and complication**

Let s be an object or system:

- **Kolmogorov complexity C(s): length of the shortest algorithm that, with no input, outputs s**
- **s is complex: Kolmogorov complexity is roughly as long as s, i.e. C(s)≈|s|**

  => there are no short descriptions of s.
- **s is complicated: its description d(s) is much longer than its Kolmogorov complexity, i.e. |d(s)|>>C(s).**

Example:
The object

b=0101010101010101010101010101 ... 0101010101

of length 200 bits has small complexity.
A generating algorithm

```
A: for i:=1 to 100 write('01')
```

is short: C(b)≤28=|A|

A' is complicated:

```
A': write('01010101010101010101010101010101');
    write('01010101010101010101010101010101');
    write('01010101010101010101010101010101');
    write('01010101010101010101010101010101');
    write('01010101010101010101010101010101');
    write('01010101010101010101010101010101');
    write('01010101010101010101').
```

|A'|=270>>C(b).

Example:

The random object

b'=1101001011101000100101100110 … 1011010111

of length 200 bits is complex.

A generating algorithm has to enumerate each and every single bit

```
write('1101001011');
write('1010001001');
write('01100110 …');
…
write('1011010111')
```

i.e. C(b')≈|b'|.

3 Orthogonalization as a fundamental idea of informatics

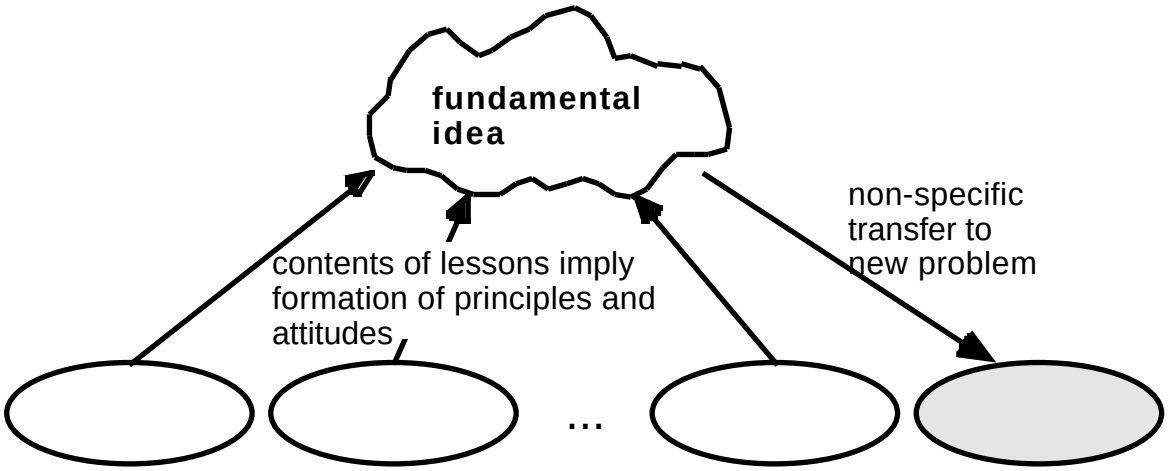
Background:

Jerome Bruner's educational psychology

applied to informatics lessons

Key notion: non-specific transfer

- **long-term (often life-long) effect**
- **you learn fundamental notions, principles, and ways of thinking (fundamental ideas)**
- **you develop views and attitudes, e.g. to learning itself, to research, to science, to conjectures, heuristics, to your own achievements etc.**
- **you consider problems occuring later as special cases**
- **you develop a cognitive meta-level**



Definition

A fundamental idea of computer science is a schema for thinking, acting, describing or explaining which satisfies four criteria:

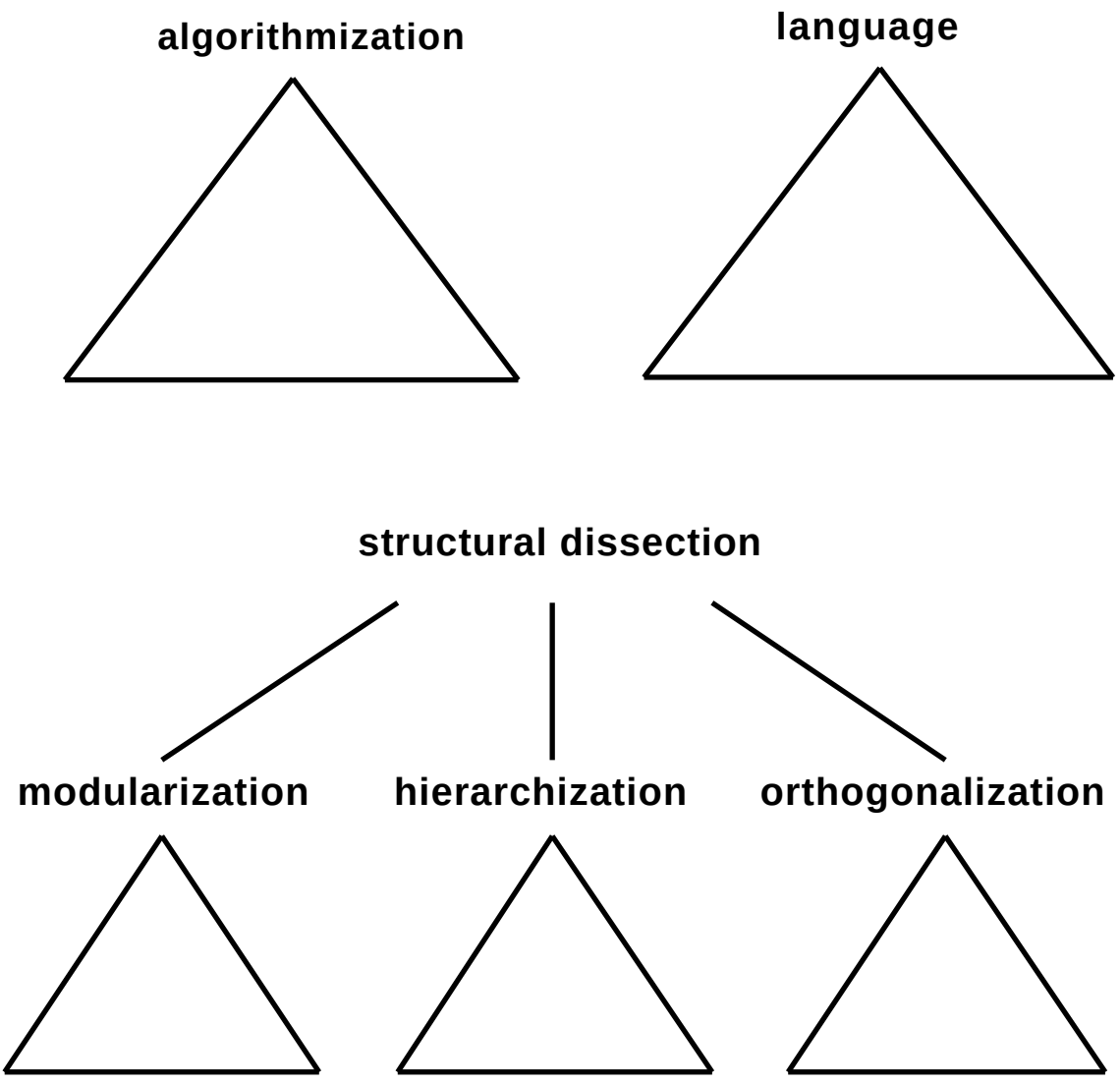
- **The Horizontal Criterion**
  - **applicable or observable in multiple ways and in different areas of computer science**
  - **organizes and integrates a wealth of phenomena**

- **The Vertical Criterion**
  - **may be taught on every intellectual level**
  - **presentations differ only by level of detail and formalization**

- **The Criterion of Time**
  - **can be clearly observed in the historical development of computer science**
  - **will still be relevant in the long run**

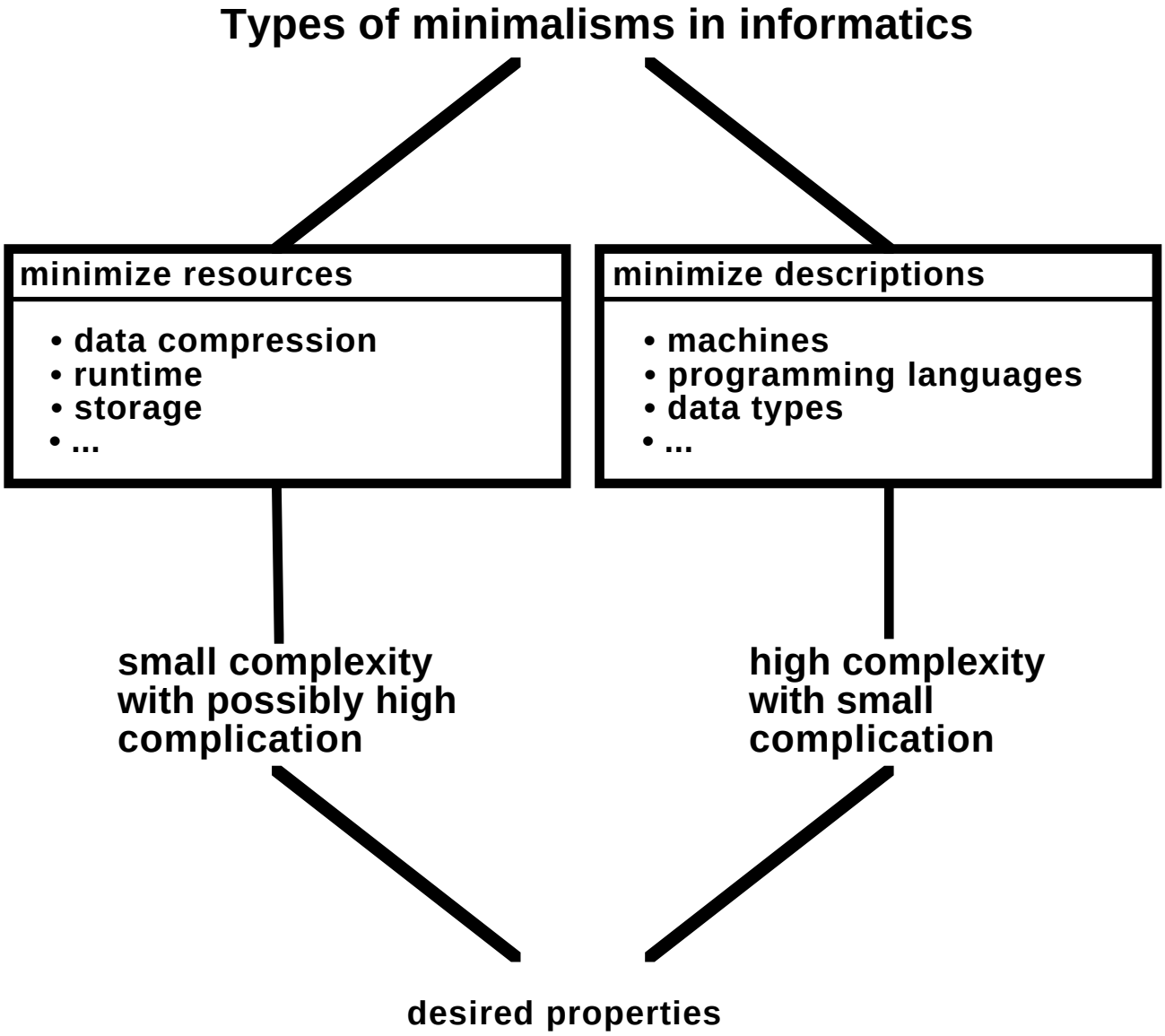
- **The Criterion of Sense**
  - **has meaning in everyday life**
  - **is related to ordinary language and thinking**

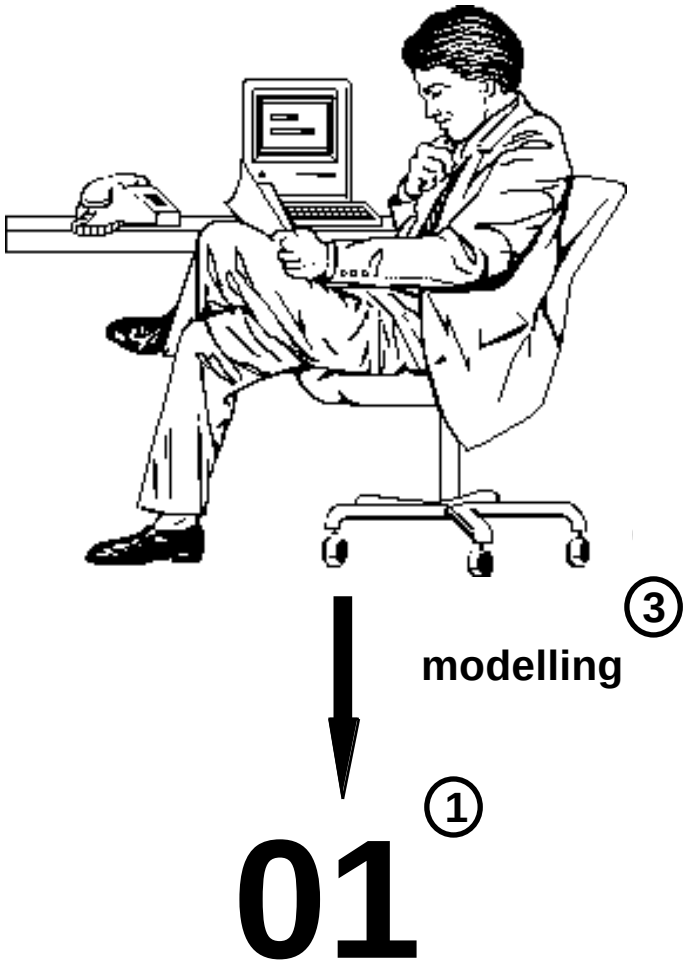
Fundamental ideas of informatics

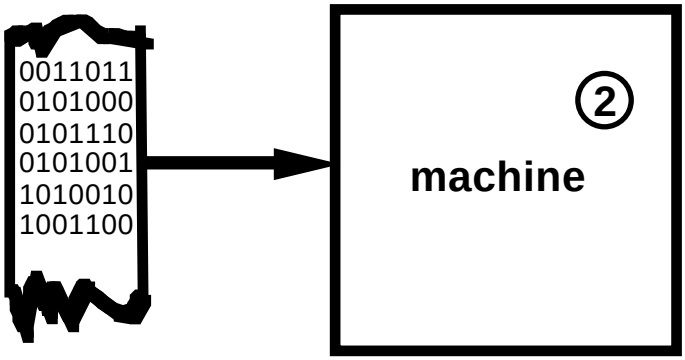


-> Already shown for orthogonalization: Criterion of Sense

4 Orthogonalization and minimalism in informatics







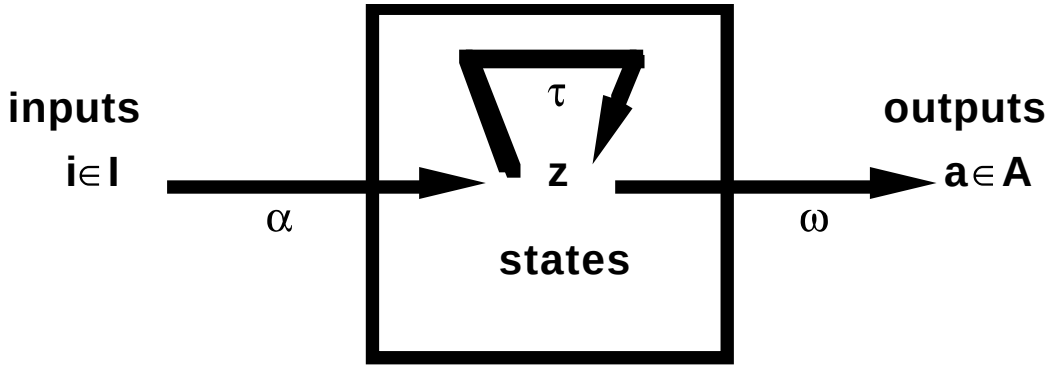
---> find construction kits

A Machines

Objective [Minsky, Thue, Church, Turing, ...]

- **What is the nature/the essence of an automaton?**
- **What are the minimal properties an automaton has to have to be called an automaton?**

General model of an automaton



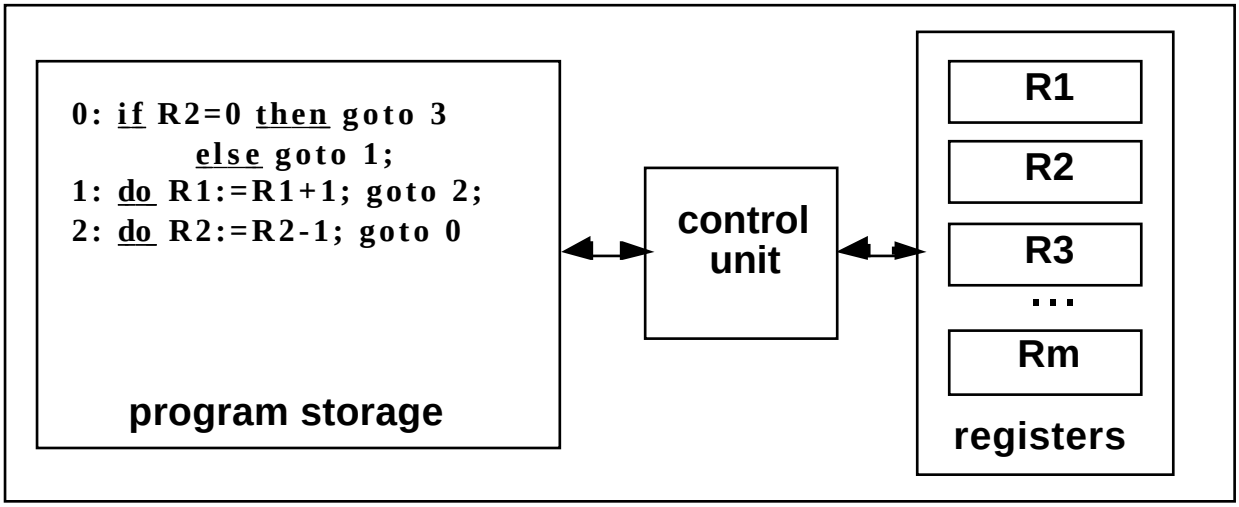
Problem: no limitations to α, τ, ω.

Consequently:
• reduce to what's feasible
• do not delete what's feasible

Orthogonalization has led to an automaton where freedom has been exactly restricted to what is feasible:

Register machine

Architecture



registers Ri contain arbitrary natural numbers

Operations

- **register operations +1 and -1**
- **zero-test Ri=0?**
- **i: do f; goto j** **(f register operation)**
- **i: if t then goto j else goto k** **(t zero-test)**

Orthogonalization in excess

- **two registers are enough (smart coding of registers into one)**
- **one register operation is enough:**

  sub 1 and jump to <label> if negative

B Control structures

Minimal construction kits for imperative control structures

- **Kit 1:** **({assignment, goto}, {concatenation, if_then_else_fi})**

- **Kit 2:** **({assignment}, {concatenation, while_do_od, if_then_else_fi})**

Collatz problem (L. Collatz, 1937)

Complexity of kit 2 is best experienced by the following simple program:

```
var x: integer;
read(x);
while x>1 do
    if even(x) then x:=x/2 else x:=3*x+1 fi
od;
write("I am done").
```

Does this program terminate for all inputs?

--> open problem for more than 60 years

C Functional programming – λ-calculus

Classical superminimal system to define and use functions given by algorithms

Let X be a set of variables.

λ-terms

1. **Every $x \in X$ is a λ-term**

2. **M and N are λ-terms => (MN) is a λ-term.**
   Application of term M to term N, usually written as M(N).

3. **If $x \in X$ is a variable and M a λ-term => (λx.M) is λ-term.**
   - **Abstraction of a λ-term M to a function λx.M**
   - **x formal parameter**
   - **M function body**
   - **λ stands for keyword function**
   - **dot separates function head and function body**
   - **effect: parametrization of an expression like $M \approx x+5$ and transfer to a function $(\lambda x.M) \approx f(x)=x+5$**
   - **λ-terms are not identified by names.**

Example

((λx.((xd)((λy.(xy))a)))b)

Application

β-rule ((λx.M)N)=M[x←N]

Examples

1. **constant function (λx.z): ((λx.z)y) -> z for any y**
2. **identity function (λx.x): ((λx.x)y) -> y for any y**

Superminimal but supercomplex

λ-calculus is a fully-functional programming language (like ML, Lisp, Haskell)

Proof: Simulate register machine

- **natural numbers**

  0 by the λ-term (λf.(λx.x)),

  1 by the λ-term (λf.(λx.(fx)))

  2 by the λ-term (λf.(λx.(f(fx))))

  3 by the λ-term (λf.(λx.(f(f(fx)))))

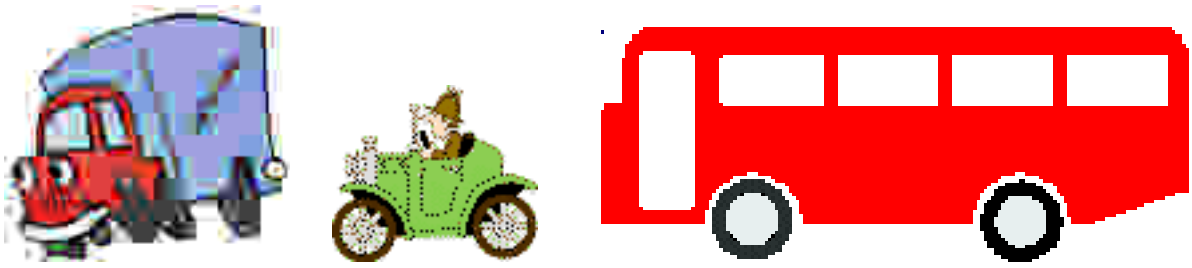
- **addition +1**

  (λn.(λf.(λx.(f(n(fx))))))

Ex: (λn.(λf.(λx.(f(n(fx)))))) (λf.(λx.(fx))) -> (λf.(λx.(f(fx))))

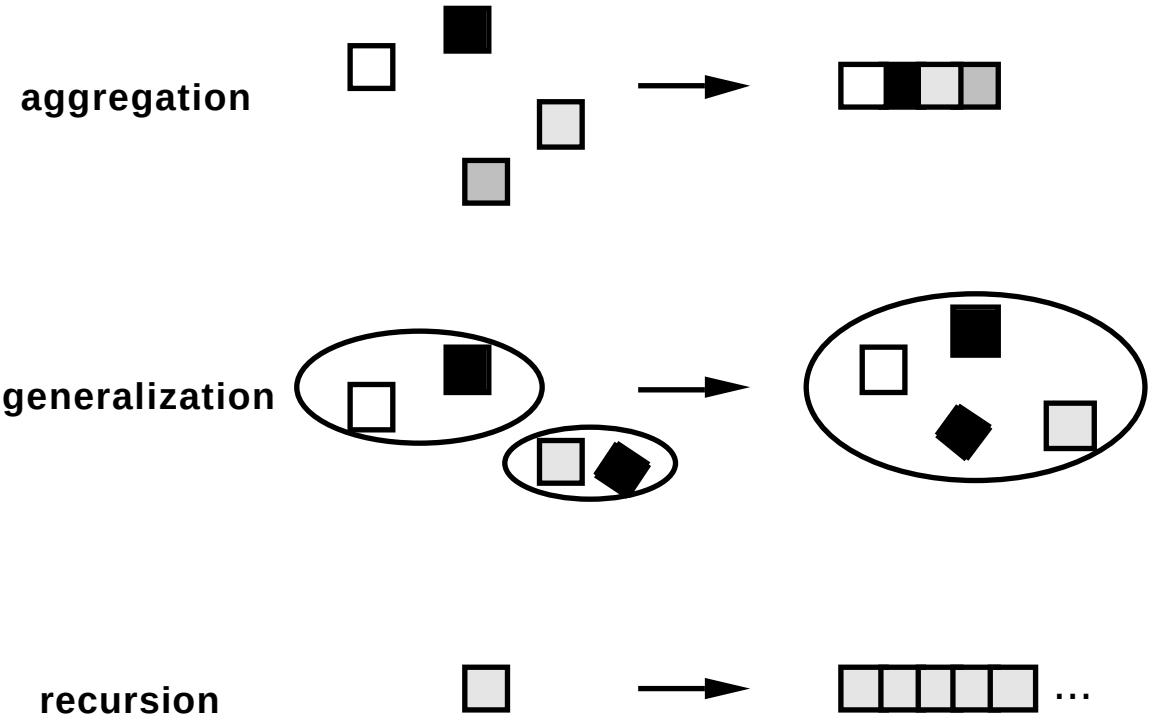
D Modelling – data types

Real world: A traffic jam



Construction kit to map any real world situation into a computer

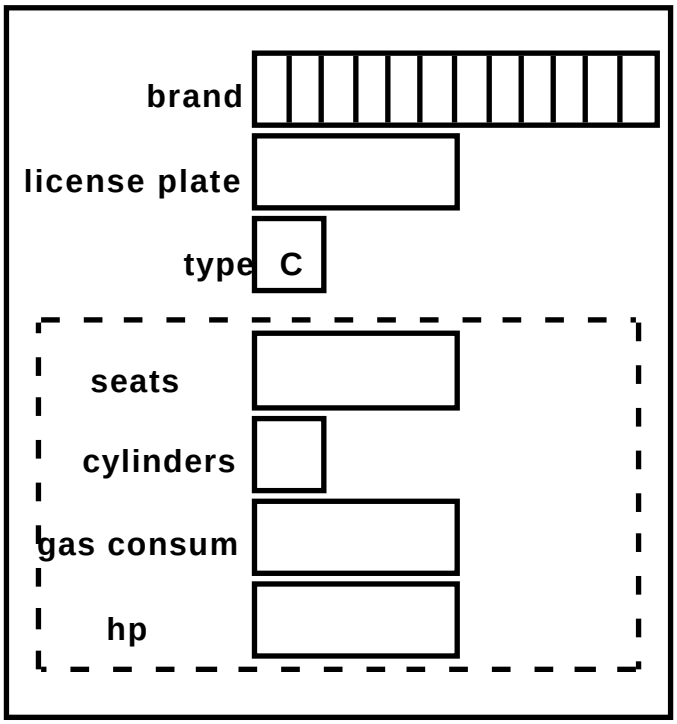
- **basic elements:** character, integer, real, boolean
- **constructors:** aggregation, generalization, recursion, functional spaces



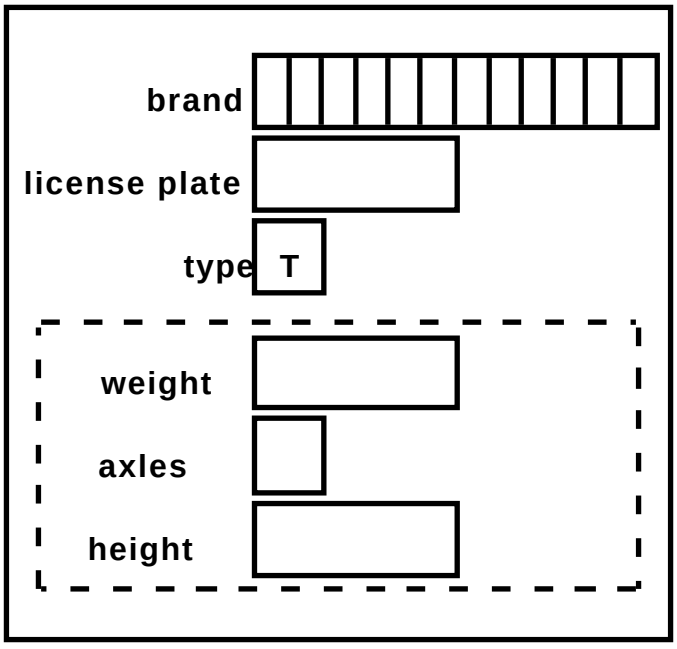
Example:

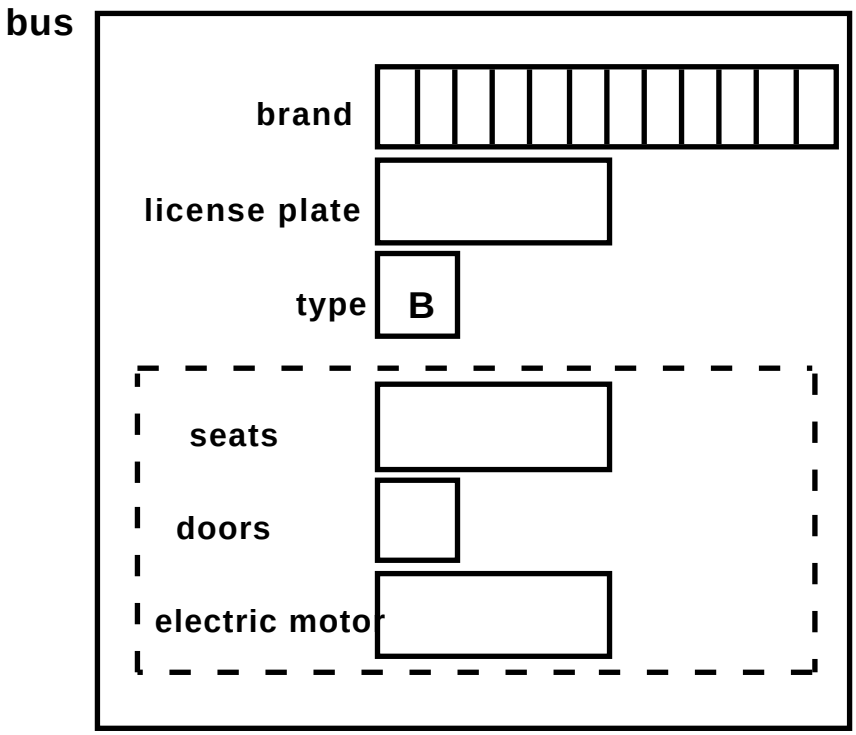
1. vehicle=car ∪ truck ∪ bus

car



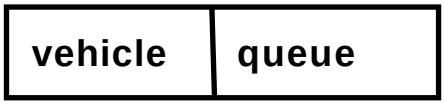
truck

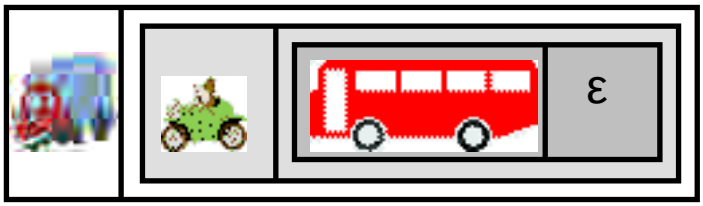




2. queue = sequence of vehicles

queue = ε ∪





E Turing machines

- **Minimal system: the universal Turing machine**

  = one-element basis of the class of all Turing machines or of the class of computable functions, resp.

- **complexity of the basis is best experienced by a Busy-Beaver-Turing machine**

F Object-oriented programming

- **Minimal system attempted: collection of reusable, reconfigurable modules which can be combined to solve any problem for any application area**

- **Minimal system attempted in object-oriented design: design patterns (e.g. the model-view-controller pattern) that form an abstract language to describe solutions to recurring object-oriented design problems**

G Boolean functions

- **Minimal systems: {and, or, not} or {and, not} or {nand} or {nor}**
- **Post/Yablonsky: five simple criteria for a set to be a basis**

H ... Z Many more ...

5 Minimal systems in informatics lessons

What we have found:

- **Orthogonalization is a fundamental idea of informatics.**
- **Orthogonalization satisfies the Horizontal Criterion and the Criterion of Sense.**

What to do:

- **The idea must definitely be integrated into school lessons**
- **Make visible its property of linking together different informatics subjects.**
- **Use its property of integrating diverse phenomena and methods under a common concept.**

How to do:

- **Teach orthogonalization on every stage of intellectual development along the spiral principle with increasing level of elaboration and formalization starting in primary school.**
- **Make clear where exactly the idea appears and applies in the concrete subject and what advantage it gives for solving the problem.**
- **Explain in detail what the current minimal system is, what its basic elements and operations are and how they work together.**
- **Analyze in more or less detail the inner structure of the field the minimal system defines.**
- **Show how this approach and earlier applications of the idea in other subjects coincide or differ.**

Key concepts: The Vertical Criterion and the spiral principle

Methodical aspects:

- **Students usually have gained early experiences with construction kits in their everyday life ("LEGO").**
- **Students may easily grasp small systems with few orthogonal basic elements and operations.**
- **Students may, along constructivism, playfully experiment with basis and operations.**
- **Students may explore, maybe in projects, the space "spanned" by the construction kit.**
- **Students may experience its possibly vast complexity.**
- **Explanations of the teacher seem hardly necessary.**

<Picture of the Smart Car>

REDUCE TO THE MAX

SLOGAN OF THE
SMART CAR COMPANY

INFORMATICS IS DONE WITH IT SOME 40 YEARS