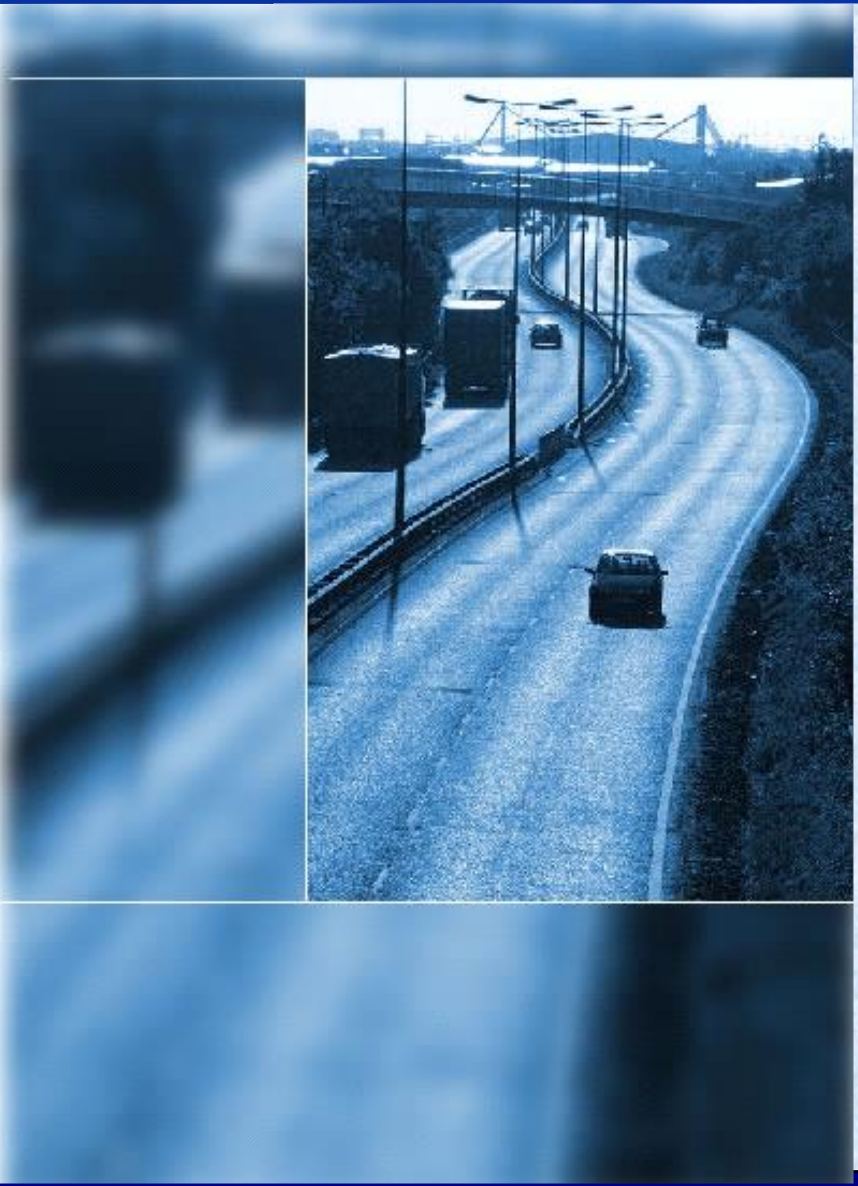
Czech Technical University in Prague



Faculty of Transportation Sciences
Department of Transport Telematics

Systems analysis
8th lecture
Graph transmission, Basics from cybernetics

Zuzana Bělinová, Zdeněk Votruba

Lecture - overview

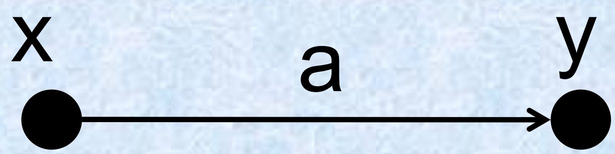
- Graph transmission – Mason-Truxal rules
- Basics from cybernetics
- Grammars

Graph transmission

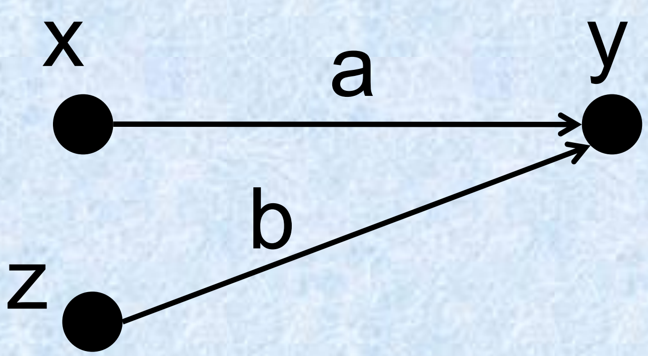
- Procedure of transferring of system of linear equation into a graph (the system matrix must be partially empty)
- It serves for simple solving of system transmission (proportion of output and input)

How to transfer the equations into graph

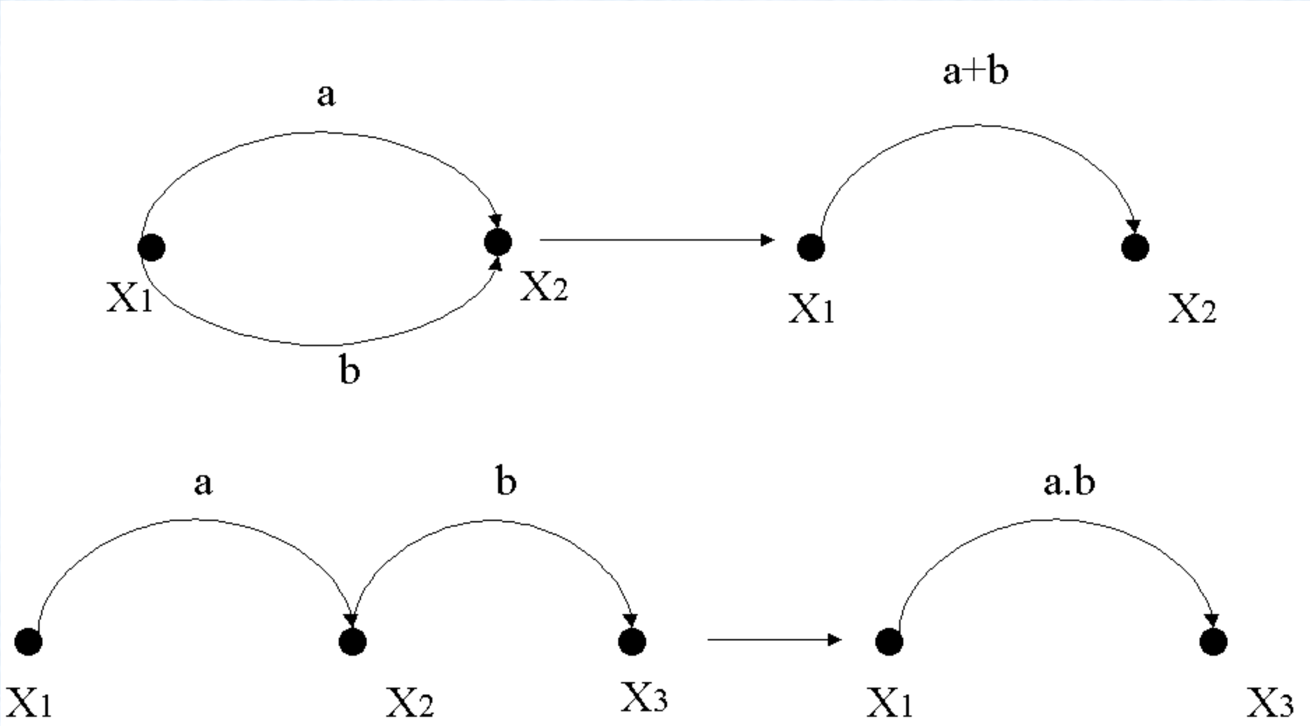
- $y=a*x$



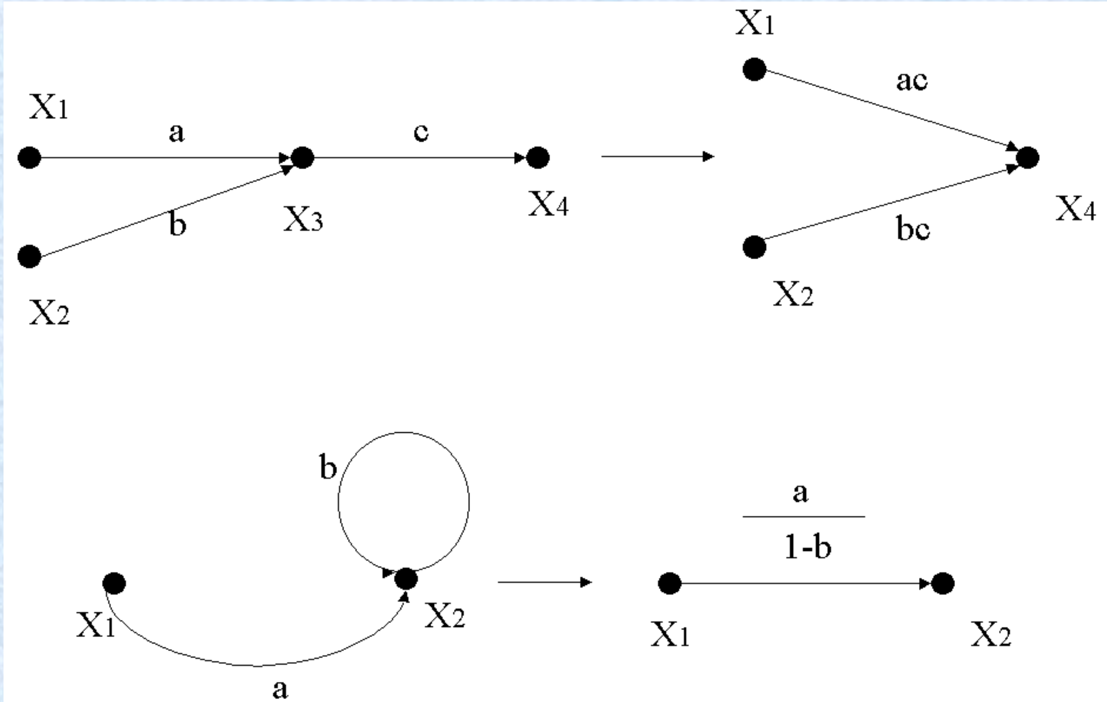
- $y=a*x + b*z$

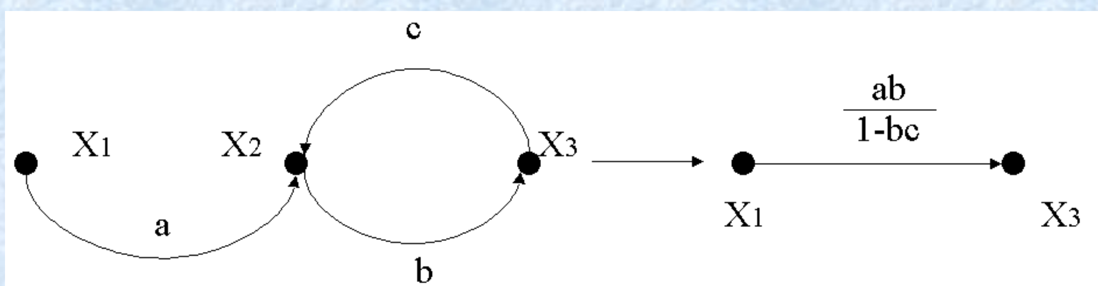


The Mason-Truxal rules I

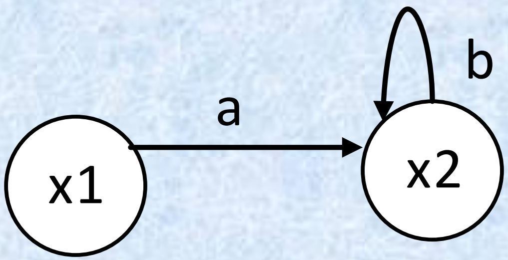


The Mason-Truxal rules II

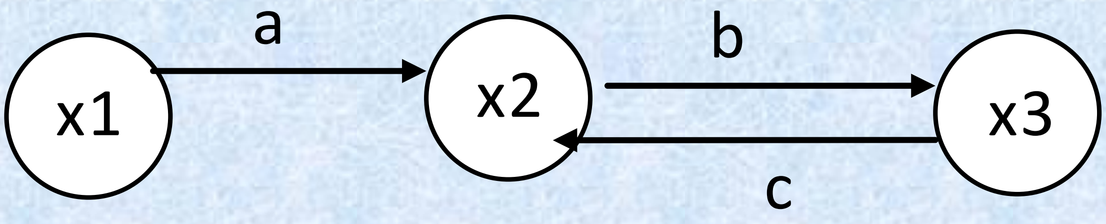




Feedback rules - deduction



- $x2=a*x1+b*x2$ x1
- $x2=a*x1+b*x2$ $x2-b*x2=a*x1$
- $x2(1-b)=a*x1$
- $x2/x1=a/(1-b)$



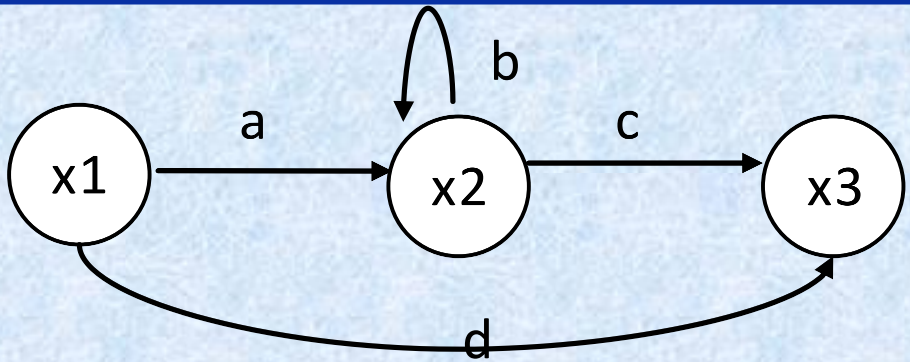
$x2=a*x1+c*x3$
$x3=b*x2$

$x2=x3/b$
$x3/b=a*x1+c*x3$
$x3/b-c*x3=a*x1$
$x3-c*b*x3=a*b*x1$
$x3(1-b*c)=x1*a*b$
$x3/x1=a*b/1-b*c$

Examples

$x_2 = a*x_1 + b*x_2$

$x_3 = c*x_2 + d*x_1$



$$\frac{x3}{x1} = \left(\frac{a}{1-b} * c\right) + d$$

- Verification

$$x_2 = \frac{a * x_1}{1-b}$$

$$x_3 = c * \frac{a * x_1}{1-b} + d * x_1$$

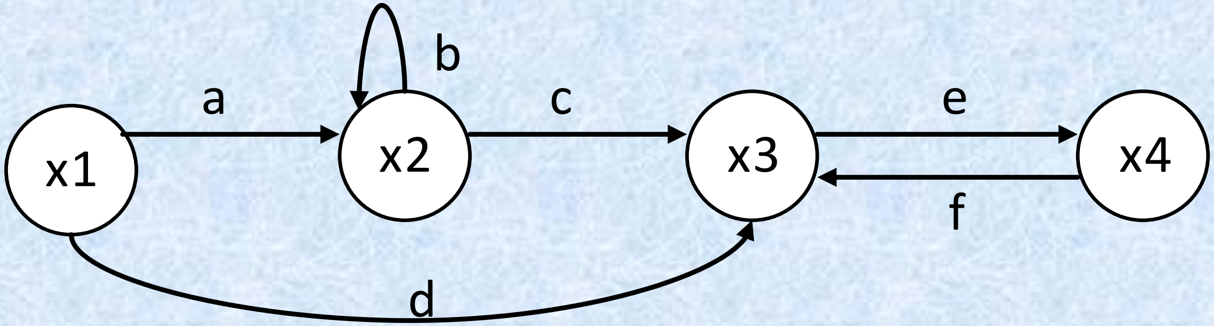
$$\frac{x3}{x1} = \frac{a * c}{1-b} + d$$

Examples

$x_2 = a*x_1 + b*x_2$

$x_3 = d*x_1 + c*x_2 + f*x_4$

$x_4 = e*x_3$



$$\frac{x_4}{x_1} = \frac{\left(\frac{a}{1-b} * c + d\right) * e}{(1 - e * f)}$$

Cybernetics Fundamentals

Cybernetics Fundamentals

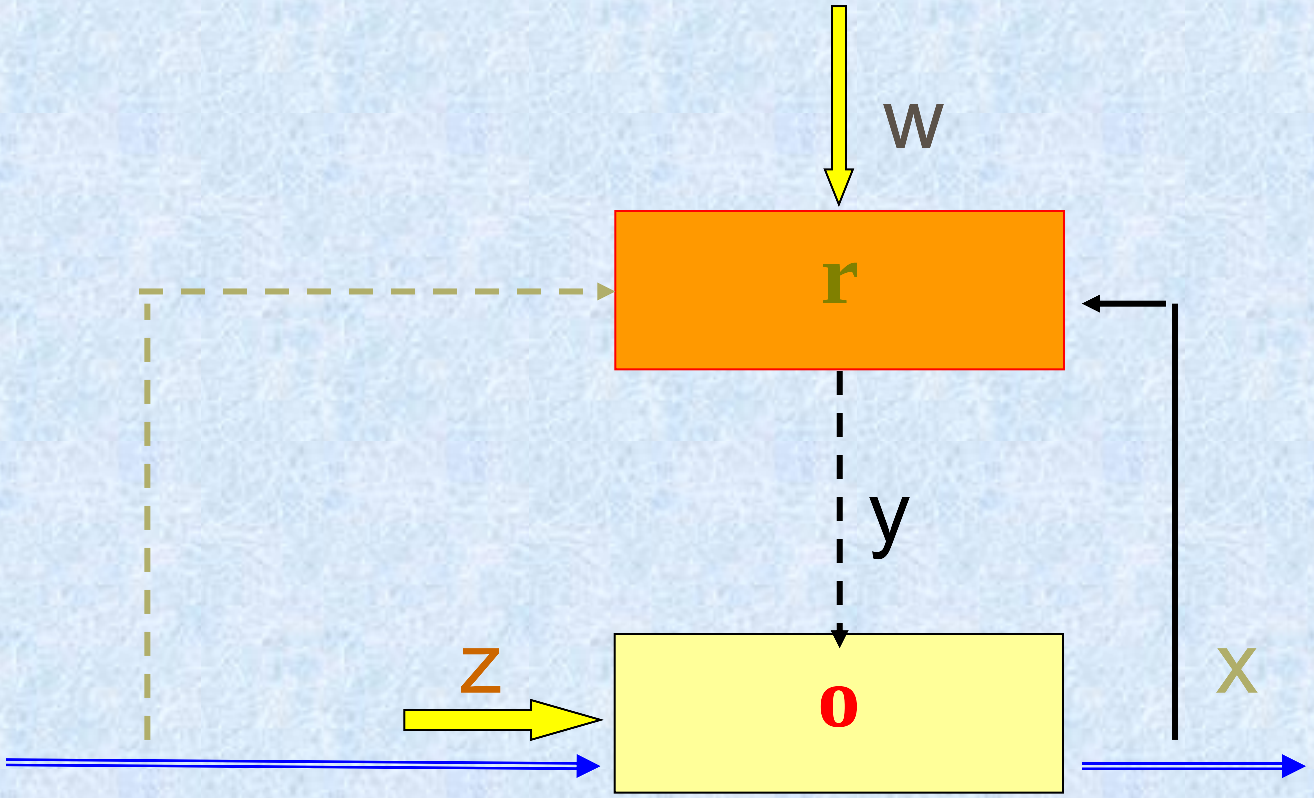
N. Wiener: Science on control and communication in living organisms and machines

Crucial role of feedback, mostly continuous variables

Control:= Consciously chosen goal seeking process

- Explicit / implicit
- Direct / Feed-back control (deviation & error) – adaptive – interactive – with learning…

Feed-back Control

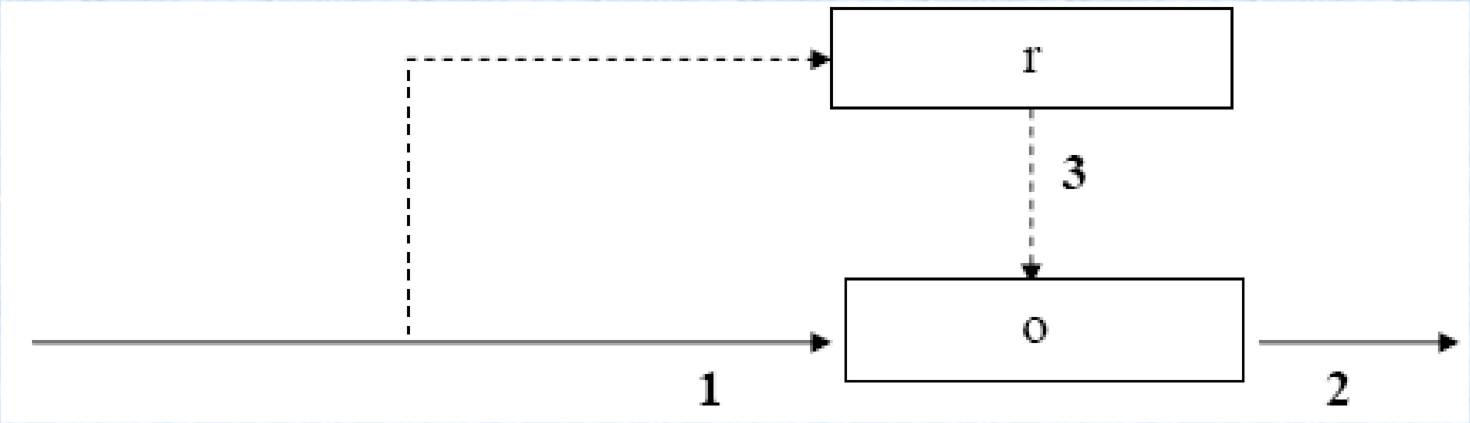


control transmission $\boldsymbol{x/w = ro/(1+ro)}$

failure transmission $\boldsymbol{x/z = o/(1+ro)}$

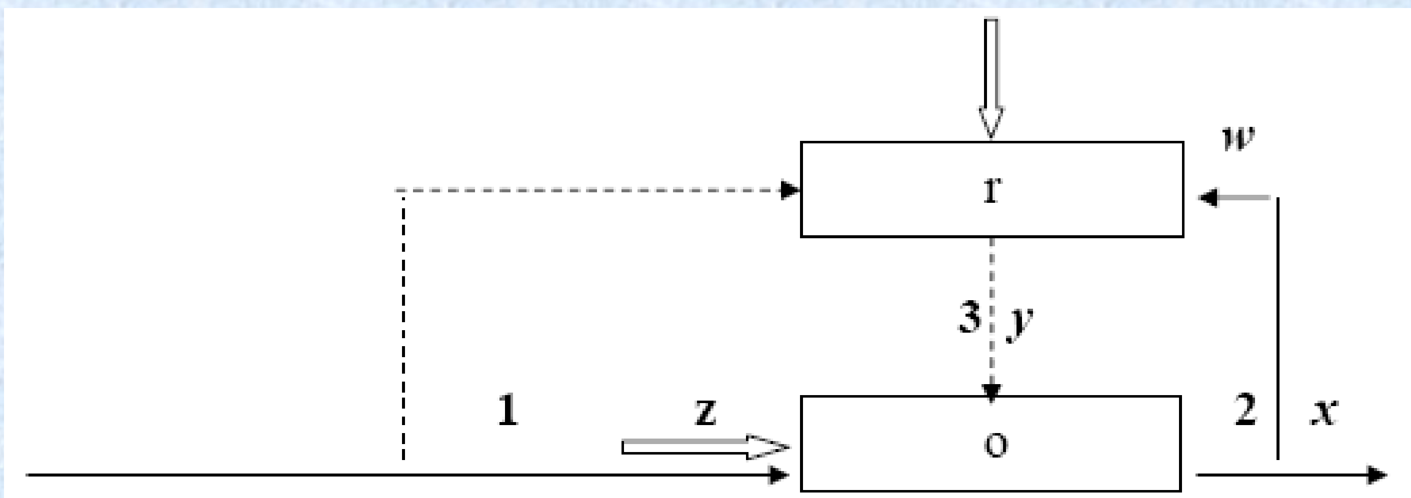
Simple control (open-loop controller)

- System does not have information about its outputs. There is no explicit feedback.
- This type of control is used only when output is robust, safe – e.g. two state control (binary)



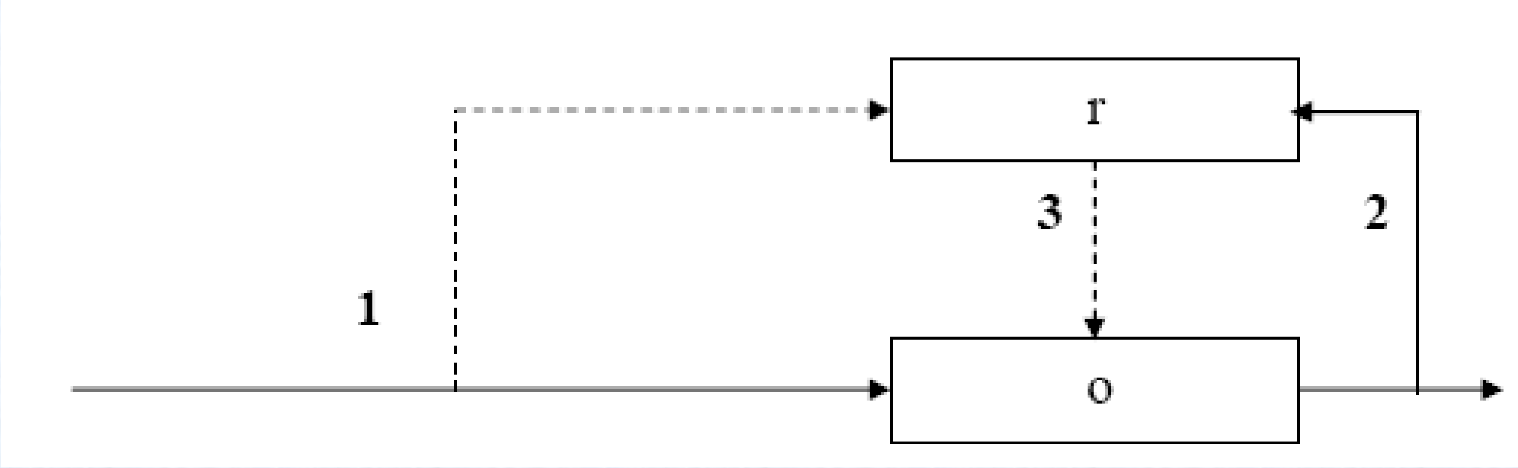
Control according variation (close-loop control)

- Basic and the simplest type of feedback control. The controller „r“ is influenced by output „2“ (with value „x“). Output value is compared to the actual reference value „w“. Depending on the variation (x – w) „r“ creates („r“ is also called the regulator) value y, transmitted on the control connection „3“ and influencing „o“.
- Note: the reference value may change in time



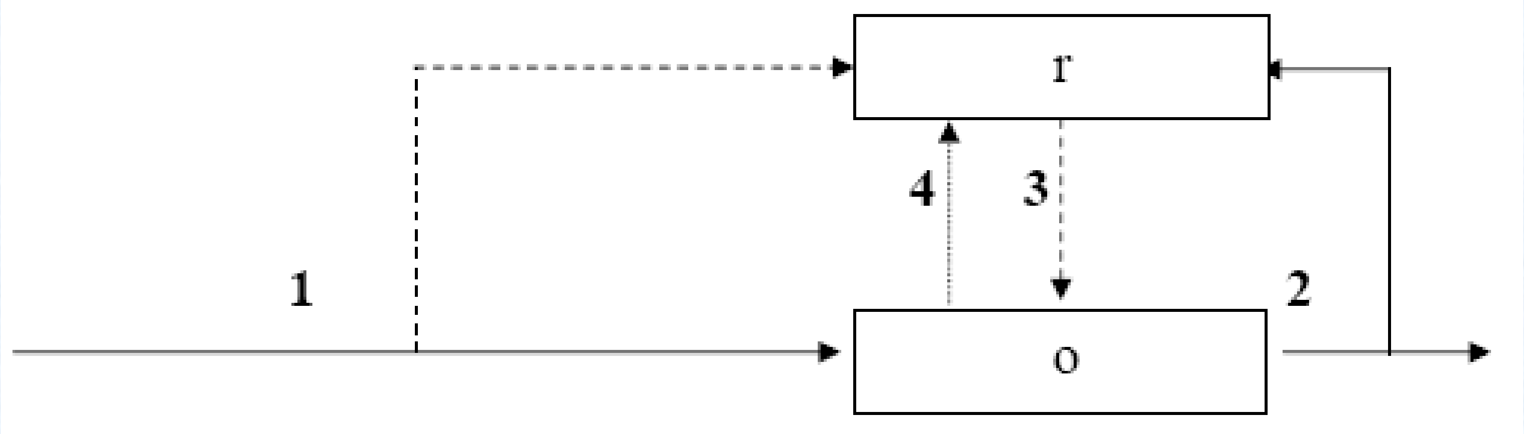
Adaptive control

- Unlike the simple feedback control function „r“ is modified using input „1“. This enables more accurate control



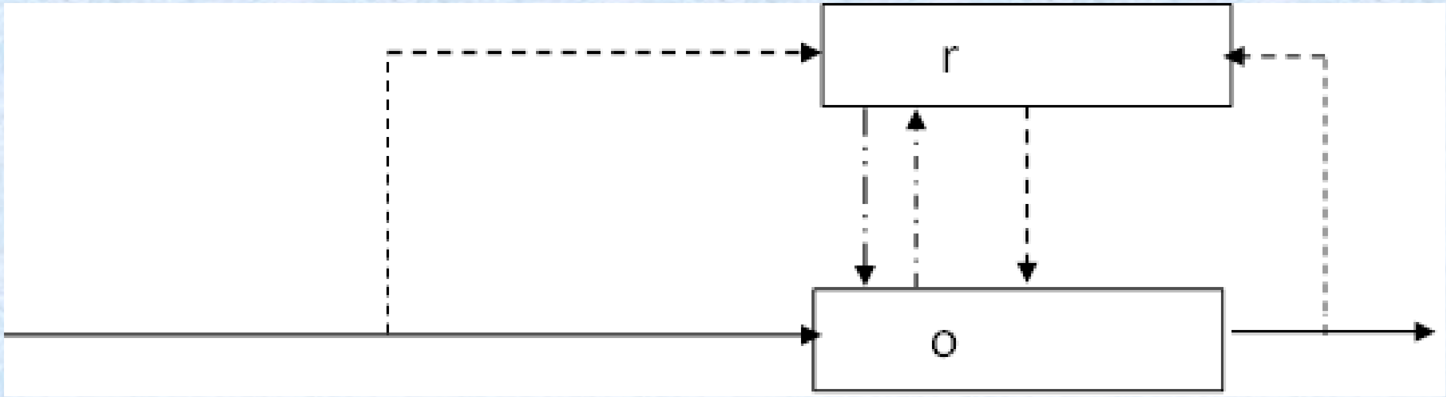
Interactive control

- A new connection is added, the ineer connection „4“. Using this connection „r“ gains information about current state „s“. This enables further improve qualitative, quantitative and dynamic parameters of the control.



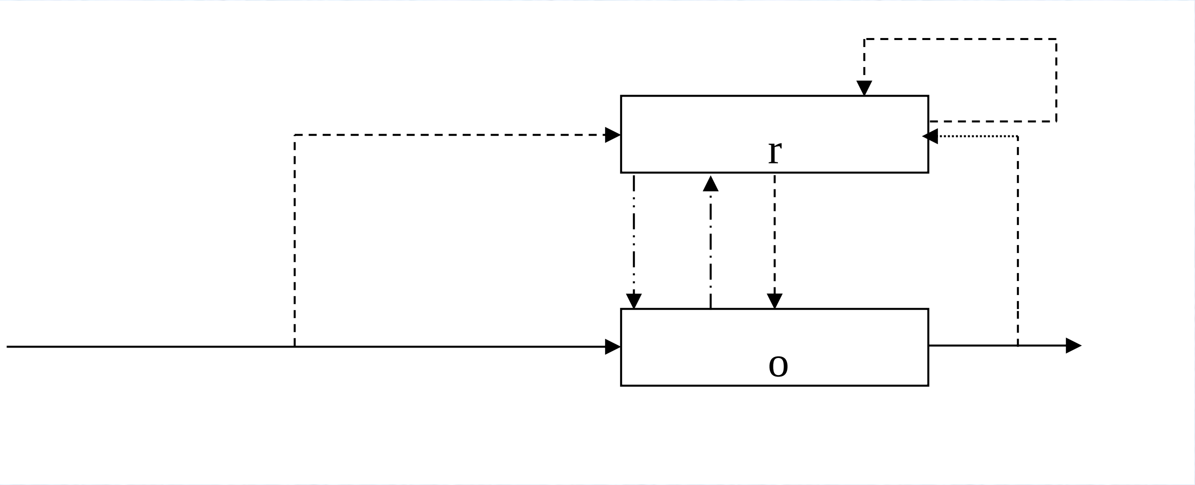
Control with premises (using genetic code)

- In the structure there is a new connection, which serves for control of the process distance from the genetic code.



Control with learning

- In the structure we can additionally identify loop in the control subsystem „r“. This means (in the simplified form) the possibility of storing information about control process in time and gaining experience from this.



Active filtration

(Ross Ashby)

- Control lowers uncertainty ⇔ Control **actively** (⇒ resources - Matter,Energy,Information consumption) **filtrates** (external) impacts.

<u>Active filtration is not effective if</u>:

1. There is shortage of Systems resources
2. Impact causes failure / destruction
3. **There is deficit of information on changes of impacting factors ⇔ unknown dynamics of environment**

Limit of active filtration - Law of requisite variety

Uncertainty removed via control is upper bounded by the median shared **information** between the controller (r) and environment.

- Logic Systems → CS (informatics)
- Communication: Forms & Transforms of Information

→**Languages**

→**How Information activates process**

Directly – reflexive level

Indirectly – via langue constructs

Event→ language construct→ interpretation→ activation

Language

- Natural language
- Programming language
- Pragmatic language

<u>Language components</u>:

1. Alphabet
2. Syntax
3. Semantics

Grammar

$G := (N, T, W, P)$

- Chomski types of grammars 0-3

Grammars

Grammar

- Grammar is foursome G=(N, T, P, S) where:
  - *N* - set of non-terminal symbols
  - *T* - set of terminal symbol
  - *P* - set of rules
  - *S* - grammar starting symbol ($S \in N$)

Rules

- Set of rules

$$P \subset (N \cup T)^* N \, (N \cup T)^* \times (N \cup T)^*$$

$(N \cup T)^*$ is any string of terminal and non-terminal symbols

- rule $(\alpha,\beta) \in P$ is written as $\alpha \rightarrow \beta$

  the meaning is: „ rewrite α to β“

- On the left side of the rule there is always a non-terminal symbol (i.e. using the rule it is always possible to rewrite some non-terminal symbol)

Example of simple grammar

- Grammar generating symetric strings of zeros and ones
  0000…01…11111
- $G = (N,T,P,S)$

  $N = \{ S, A \}$

  $T = \{ 0, 1 \}$

  $P = \{ S \rightarrow 0A1,\ A \rightarrow 0A1,\ A \rightarrow \varepsilon \}$

  (symbol ε means empty symbol)

- Example of generated string (sentence):

  $S \rightarrow 0A1 \rightarrow 00A11 \rightarrow 000A111 \rightarrow 000111$

Chomski types of grammars 0-3

- According rule types
  - Type-0 grammars (unrestricted grammars)
  - Type-1 grammars (context-sensitive grammars)
  - *Type-2 grammars (context-free grammars)*
  - *Type-3 grammars (regular grammars)*
- Unrestricted – all formal grammers
- Context-sensitive
  - $\gamma 1 A \gamma 2 \rightarrow \gamma 1 \beta \gamma 2$, $A \in N$, $\gamma 1, \gamma 2$ is context, $\beta \in (N \cup T)+$
- Context-free
  - $A \rightarrow \beta$, $A \in N$, $\beta \in (N \cup T)+$
- Regular
  - A $\rightarrow$ aB or A $\rightarrow$ a, where A, B $\in$ N, a $\in$ T

Thank you for your attention

Semantics; relation: Language ⇔ Automaton

- Axiomatic
- Compiler oriented (Translation between languages)
- Logic (Semantics of construct is generated as the output of certain logic functions)
- Denotation based (reality of use)

Isomorphism between language and automaton→ epistemological consequences :

1. System can be recorded as a set of languages and corresponding set of rules of their mutual translation
2. Real object can be expressed as a construct in certain language.

- Principle of homeostasis
- Ordering; IInd law of thermodynamics ; self-ordering
  - <u>Tasks of AI</u>
    (i.) identification
    (ii.) optimization
    (iii.) definition of goals
    (iv.) (self) adaptibility
    (v.) control of parallel processes
    (vi.) communication with environment
    (vii.) understanding;
    (viii.) control of identity
    (ix.) (self) consciousness