Peter Naur's contribution to formal notations and beyond

Viktor Gsteiger

University of Basel
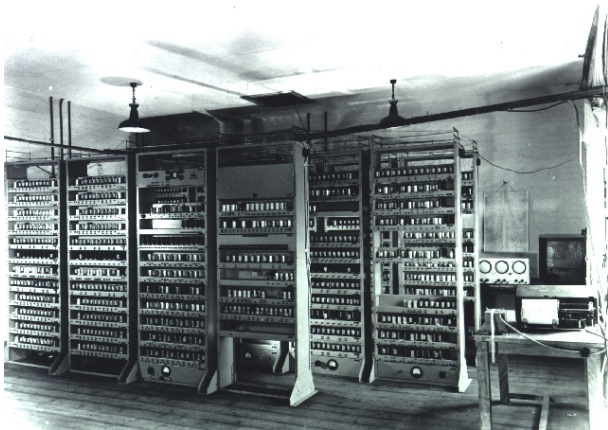
Seminar Turing Award Winners and Their Contributions
v.gsteiger@unibas.ch

November 26, 2020

Peter Naur's contribution

Citation

For fundamental contributions to programming language design and the definition of Algol 60, to compiler design, and to the art and practice of computer programming.

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
●○○○   ○○○○○   ○○   ○○○
○○○○○○   ○○○○○○○○
○○   ○○○

Phrase structure grammars

Phrase structure grammar

Definition

Quadruple $G = (V, \Sigma, P, S)$, where

- V is vocabulary
- $\Sigma \subseteq V$ *terminal symbols*
- $S \in (V - \Sigma)$ *start symbol*
- $P \subseteq V^*NV^* \times V^*$ *productions*

Formal notations | Peter Naur's contribution | Computing vs. Human Thinking | Conclusion
○●○○ | ○○○○○ | ○○ | ○○○
○○○○○○ | ○○○○○○○○
○○ | ○○○

Phrase structure grammars

Phrase structure

Example

$$G = (\{S, NP, VP, V, \text{the man}, \text{the book}, \text{took}\},$$
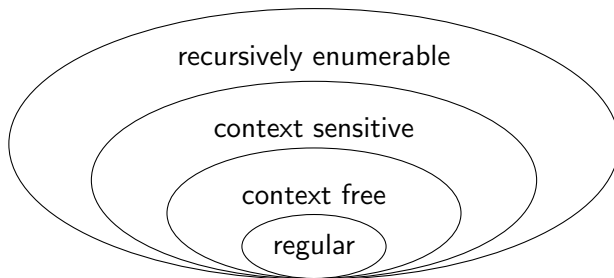$$\{\text{the man}, \text{the book}, \text{took}\}, P, S)$$

$$P:$$
$$S \rightarrow NPVP$$
$$VP \rightarrow VNP$$
$$NP \rightarrow \text{the man}$$
$$NP \rightarrow \text{the book}$$
$$V \rightarrow \text{took}$$

Formal notations | Peter Naur's contribution | Computing vs. Human Thinking | Conclusion

Phrase structure grammars

Chomsky Hierarchy

recursively enumerable

context sensitive

context free

regular

Formal notations | Peter Naur's contribution | Computing vs. Human Thinking | Conclusion
0000 | 00000 | 00 | 000
000000 | 00000000
00 | 000

Phrase structure grammars

Phrase structure in tree form

Example

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion

○○○○   ○○○○○   ○○   ○○○
●○○○○○   ○○○○○○○○
○○
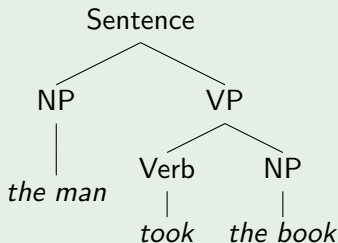
Backus Naur form

Backus Normal form

Example

$$< S > :\equiv < NP > \quad < VP >$$
$$< VP > :\equiv < V > \quad < NP >$$
$$< NP > :\equiv \text{the man } or \text{ the book}$$
$$< V > :\equiv \text{took}$$

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
OOOO   OOOOO   OO   OOO
O●OOOO   OOOOOOOO
OO   OOO

Backus Naur form

Power of Backus Naur Form

- BNF only generates exactly the context free languages
- EBNF has Exceptions (R - E) $\rightarrow$ not context free

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
○○○○   ○○○○○   ○○   ○○○
○○●○○○   ○○○○○○○○
○○   ○○○

Backus Naur form

Theorem

Production rules of the form R - E have no BNF equivalent

Proof.

Let $L_1 = \{a^n b^n a^m | n, m \geq 0\}$ be generated by $\langle L_1 \rangle ::= \langle X \rangle \langle A \rangle$
$\langle X \rangle ::= a \langle X \rangle b | \varepsilon$ $\langle A \rangle ::= \langle A \rangle a | \varepsilon$
Let $L_2 = \{a^n b^m a^m | n, m \geq 0\}$ be generated by $\langle L_2 \rangle ::= \langle A \rangle \langle X \rangle$

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
oooo   ooooo   oo   ooo
oooo●oo   oooooooo
oo   ooo
Backus Naur form

Theorem

Production rules of the form R - E have no BNF equivalent

Proof (Cont.)

Now show $L_3 = L_1 \cap L_2 = \{a^n b^n a^n | n \geq 0\}$ by pumping lemma and contradiction.

For given p $\geq$ 1 we can choose p such that s $= a^p b^p a^p \in L_3$. PL tells us $s = uvwxy$ with substrings u, v, w, x, and y, such that $|vx| \geq 1$, $|vwx| \leq p$, and $uv^i wx^i y \in L_3$. Now *uvx* can not contain more than two distinct symbols. Thus $uv^2 wx^2 y \notin L_3$.

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
○○○○   ○○○○○   ○○   ○○○
○○○○●○   ○○○○○○○○
○○   ○○○

Backus Naur form

Theorem

Production rules of the form R - E have no BNF equivalent

Proof (Cont.)

Thus $L_3 = \{a^n b^n a^n | n \geq 0\} = L_1 - (L_1 - L_2)$ is not context free. Thus R - E can not be modelled in BNF, since BNF generates only context free languages. □

Formal notations	Peter Naur's contribution	Computing vs. Human Thinking	Conclusion

Backus Naur form

- EBNF is more powerful than BNF
- BNF is exactly as powerful as context free grammars
- EBNF suffers from Russel-like paradoxes, e.g. $xx =$ "A" - xx;

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
0000   00000   00   000
000000   00000000
●0   000

Programming languages, natural languages and mathematical languages

Comparison of levels of formalization

- Natural languages
- Mathematical languages
- Programming languages

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
0000   00000   00   000
000000   00000000
00●   000
Programming languages, natural languages and mathematical languages

Language hierarchy

Formal notations
oooo
oooooo
oo
Algol in History

Peter Naur's contribution
●oooo
oooooooo
ooo

Computing vs. Human Thinking
oo

Conclusion
ooo

Historical context

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion

Algol in History
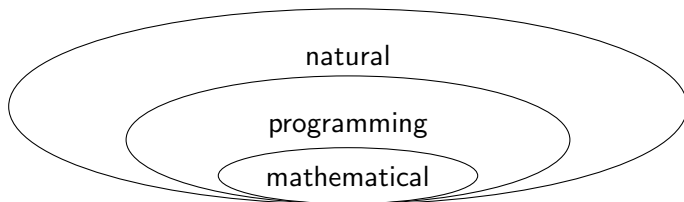
Fortran

GENERAL FORM	EXAMPLES
1 to 5 decimal digits. A preceding + or − sign is optional. The magnitude of the constant must be less than 32768.	3 +1 − 28987

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
0000   00●00   00   000
000000   00000000
00   000

Algol in History

Algol 58

3.31 Integers and numbers

$\langle\text{digit}\rangle : \equiv$ 0 *or* 1 *or* 2 *or* 3 *or* 4 *or* 5 *or* 6 *or* 7 *or* 8 *or* 9

$\langle\text{integer}\rangle : \equiv \langle\text{digit}\rangle$ *or* $\langle\text{integer}\rangle \langle\text{digit}\rangle$

$\langle\text{dn}\rangle : \equiv \langle\text{integer}\rangle$ *or* $\langle\text{integer}\rangle$ *or* $\langle\text{integer}\rangle$ *or* $\langle\text{dn}\rangle \langle\text{integer}\rangle$

$\langle\text{si}\rangle : \equiv +\langle\text{integer}\rangle$ *or* $-\langle\text{integer}\rangle$ or $\langle\text{integer}\rangle$

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
○○○○   ○○○●○   ○○   ○○○
○○○○○○   ○○○○○○○○○
○○   ○○○

Algol in History

Algol 60

```
⟨digit⟩ ::= 0|1|2|3|4|5|6|7|8|9
     Digits are used for forming numbers, identifiers, and strings.
⟨unsigned integer⟩ ::= ⟨digit⟩|⟨unsigned integer⟩⟨digit⟩
⟨integer⟩ ::= ⟨unsigned integer⟩|+⟨unsigned integer⟩| -⟨unsigned integer⟩
```

$+$ Additional examples, semantics and types

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
○○○○   ○○○○●   ○○   ○○○
○○○○○○   ○○○○○○○○○
○○   ○○○

Algol in History

Algol 68

b) An arithmetic value has a "length number", i.e., a positive integer characterizing the degree of discrimination with which the value is kept in the computer. The number of integers (real numbers) of given length number that can be distinguished increases with the length number up to a certain length number, the number of different lengths of integers (real numbers) {10.1.a,c}, after which it is constant.

c) For each pair of integers (real numbers) of the same length number, the relationship "to be smaller than" is defined {10.2.3.a, 10.2.4.a}. For each pair of integers of the same length number, a third integer of that length number may exist, the first integer "minus" the other one {10.2.3.g}. Finally, for each pair of real numbers of the same length number, three real numbers of that length number may exist, the first real number "minus" ("times", "divided by") the other one {10.2.4.g,l,m}; these real numbers are obtained "in the sense of

numerical analysis", i.e., by performing the operations known in mathematics by these terms on real numbers which may deviate slightly from the given ones {; this deviation is left undefined in this Report}.

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
○○○○   ○○○○○   ○○   ○○○
○○○○○○   ●○○○○○○○○
○○   ○○○

Algol 60

Properties of Algol 60

- Block scope
- Call-by-value, call-by-name and call-by-reference parameter passing
- Formal specification
- No I/O facilities

Formal notations
○○○○
○○○○○○
○○

Peter Naur's contribution
○○○○○
○●○○○○○○
○○○

Computing vs. Human Thinking
○○

Conclusion
○○○

Algol 60

Block scope

Example

```
begin
    integer X;
    X := 5;
    begin
        integer X, Y;
        X := 4;
        Y := 8;
    end
    print(X);
    Y := 12;
end;
```

Formal notations
0000
000000
00

Peter Naur's contribution
00000
00●00000
000

Computing vs. Human Thinking
00

Conclusion
000

Algol 60

Call-by-value

- Callee parameter Value of arguments given
- No modification of caller argument

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
oooo   ooooo   oo   ooo
oooooo   oooo●oooo
oo   ooo

Algol 60

Call-by-name

- Pass thunk
- Similar to call-by-reference
- Parameter is passed without evaluation

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
0000   00000   00   000
000000   00000●000
00   000

Algol 60

Call-by-name

Example

```
procedure swap (a, b);
integer a, b, temp;
begin
    temp := a;
    a := b;
    b:= temp
end;
```

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
○○○○   ○○○○○   ○○   ○○○
○○○○○○   ○○○○○●○○
○○   ○○○

Algol 60

Call-by-name

Example

swap(x, y):

```
temp := x;
x := y;
y := temp
```

sqap(i, x[i]):

```
temp := i;
i := x[i];
x[i] := temp
```

I/O facilities

Example

```
begin
    print('Hello world!');
    comment error!
end;
```

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
oooo   ooooo   oo   ooo
oooooo   ooooooo●
oo   ooo
Algol 60

Project

ALGOL 60 Tutorial
We shall not forget the roots of our tools

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
0000   00000   00   000
000000   00000000
00   ●00

Art and practice of computer programming

Programming as theory building

- Source code contains theory
- Transfer theory to next programmer
- Theory should be observable

Formal notations
○○○○
○○○○○○
○○

Peter Naur's contribution
○○○○○
○○○○○○○○
○●○

Computing vs. Human Thinking
○○

Conclusion
○○○

Art and practice of computer programming

Role of formal descriptions

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
0000   00000   00   000
000000   00000000
00   000

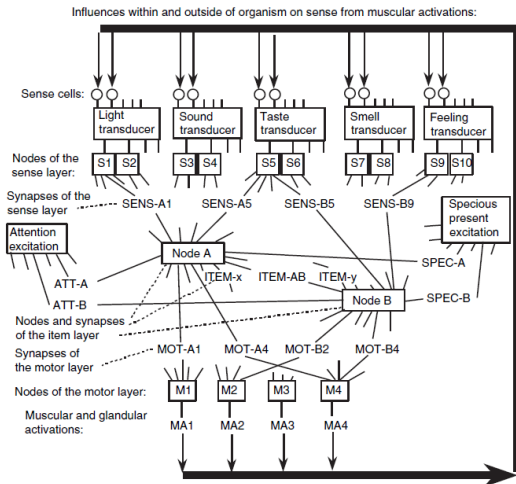Art and practice of computer programming
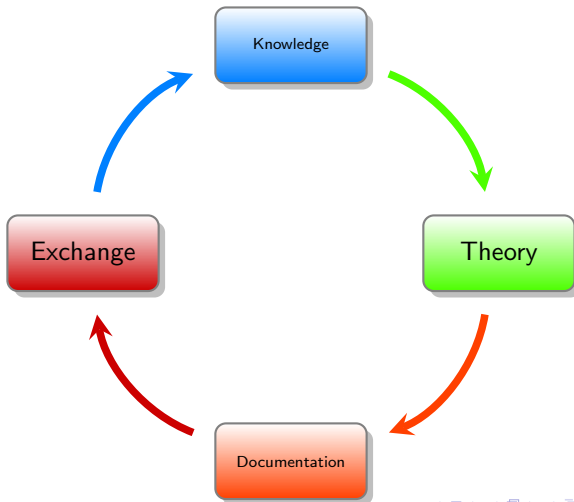
Problems of over-formalization

- Hard to exchange
- Prone to errors
- Goal in itself
- Disconnected from environment

$\rightarrow$ For an example see PROOF VERSUS FORMALIZATION by Peter Naur

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   Conclusion
oooo   ooooo   ●o   ooo
oooooo   oooooooo
oo   ooo

Computing Versus Human Thinking

- Description of computers
- Description of human thinking

Formal notations
oooo
oooooo
oo

Peter Naur's contribution
ooooo
oooooooo
ooo

Computing vs. Human Thinking
o●

Conclusion
ooo

Synapse state description

Influences within and outside of organism on sense from muscular activations:

Formal notations
○○○○
○○○○○○
○○

Peter Naur's contribution
○○○○○
○○○○○○○○
○○○

Computing vs. Human Thinking
○○

Conclusion
●○○

Importance of formal notation

For achieving clarity any formal mode of expression should be used, not as a goal in itself, but wherever it appears to be helpful to authors and readers alike.

(Peter Naur)

Formal notations Peter Naur's contribution Computing vs. Human Thinking **Conclusion**
oooo   ooooo   oo   o●o
oooooo   ooooooooo
oo   ooo

Discussion

Do you think that human thinking will be formally describable?

Formal notations   Peter Naur's contribution   Computing vs. Human Thinking   **Conclusion**
0000   00000   00   00●
000000   00000000
00   000

References

📄 Wikimedia Commons.
File:birch bark ms from kashmir of the rupavatra wellcome l0032691.jpg
— wikimedia commons, the free media repository, 2020.
[Online; accessed 8-November-2020].

📄 Wikimedia Commons.
File:edsac (19).jpg — wikimedia commons, the free media repository,
2020.
[Online; accessed 8-November-2020].

📄 Wikimedia Commons.
File:peternaur.jpg — wikimedia commons, the free media repository, 2020.

[Online; accessed 8-November-2020].