Revised Report
on the
Algorithmic Language
ALGOL 68

A. van Wijngaarden, B.J. Mailloux, J.E.L. Peck,
C.H.A. Koster, M. Sintzoff, C.H. Lindsey,

A. van. Wijngaarden... two level grammars
M. Sintzoff... nest syntax
L. Meertens... predicates
A language with two sentences

he sings a song
we sing a song

Meta rule
A) NUMBER :: singular ; plural.
Hyper-rules
a) NUMBER sentence:
   NUMBER subject, NUMBER predicate.
b) NUMBER subject: NUMBER pronoun.
c) NUMBER predicate: NUMBER verb, object.
d) object: "a song".
e) singular pronoun: "he".
f) plural pronoun: "we".
g) singular verb: "sings".
h) plural verb: "sing".
Parse tree for "he sings a song"

- singular sentence
  - singular subject,
  - singular predicate
    - singular pronoun
      - "he"
    - singular verb,
      - "sings"
    - object
      - "a song"
Metarules

A) PRONOUN :: he ; we.

B) VERB :: sing ; sings.

Hyper-rules

a) sentence : "PRONOUN", "VERB", object,

   where VERB agrees with PRONOUN.

b) object : "a song".

c) where sings agrees with he :.

d) where sing agrees with we :.
Parse tree using a predicate.

- **Sentence**
  - **Object**
    - Where
      - Sings
    - Agrees with
  - He
    - Sings
    - A song
ALGOL 68

General predicates

A) $\text{ALPHA:: } a; b; c; d; e; f; g; h; i; j; k; l; m; n; o; p; q; r; s; t; u; v; w; x; y; z.$

B) $\text{NOTION:: ALPHA; NOTION ALPHA.}$

C) $\text{NOTETY:: NOTION ; EMPTY.}$

D) $\text{EMPTY::}$

E) $\text{THING:: NOTION ; } \langle \text{NOTETY} \rangle \text{ NOTETY;}
          \text{THING } \langle \text{NOTETY} \rangle \text{ NOTETY.}$

F) $\text{WHETHER:: where ; unless.}$
Predicate hyper-rules

a) where true
b) unless false

c) where THING1 and THING2:
    where THING1, where THING2.
d) where THING1 or THING2:
    where THING1; where THING2.
e) unless THING1 and THING2:
    unless THING1; unless THING2.
f) unless THING1 or THING2:
    unless THING1, unless THING2.
Production trees for predicates

where true or false

where true

unless false or false

unless false

unless false
String matching

WHETHER \langle \text{NOTETY1} \rangle \text{ is } \langle \text{NOTETY2} \rangle :

\text{WHETHER } \langle \text{NOTETY1} \rangle \text{ begins with } \langle \text{NOTETY2} \rangle \\
\text{and } \langle \text{NOTETY2} \rangle \text{ begins with } \langle \text{NOTETY1} \rangle \text{.}
b) \text{WHETHER } \langle \text{EMPTY} \rangle \text{ begins with } \langle \text{NOTION} \rangle : \\
\text{WHETHER false.}

c) \text{WHETHER } \langle \text{NOTETY} \rangle \text{ begins with } \langle \text{EMPTY} \rangle : \\
\text{WHETHER true.}

d) \text{WHETHER } \langle \text{ALPHA1 NOTETY1} \rangle \\
\text{begins with } \langle \text{ALPHA2 NOTETY2} \rangle : \\
\text{WHETHER } \langle \text{ALPHA1} \rangle \text{ coincides with } \langle \text{ALPHA2} \rangle \\
in \langle \text{abcdefghijklmnopqrstuvwxyz} \rangle \\
\text{and } \langle \text{NOTETY1} \rangle \text{ begins with } \langle \text{NOTETY2} \rangle .
coincides with

k) where \( \langle \text{ALPHA} \rangle \) coincides with \( \langle \text{ALPHA} \rangle \)
in \( \langle \text{NOTION} \rangle \) : where true.

l) unless \( \langle \text{ALPHA1} \rangle \) coincides with \( \langle \text{ALPHA2} \rangle \)
in \( \langle \text{NOTION} \rangle \) :

where \( \langle \text{NOTION} \rangle \) contains
\( \langle \text{ALPHA1 NOTETY ALPHAA2} \rangle \)
or \( \langle \text{NOTION} \rangle \) contains
\( \langle \text{ALPHA2 NOTETY ALPHAA1} \rangle \).
contains

m) WHETHER <ALPHA NOTETY>
   contains <NOTION>:
   WHETHER <ALPHA NOTETY>
   begins with <NOTION>
   or <NOTETY> contains <NOTION>.

n) WHETHER <EMPTY> contains <NOTION>:
   WHETHER false.
ALGOL 68

Mode declarations

mode \texttt{lr} = \texttt{long real} ;

mode \texttt{vector} = [1 : n] \texttt{real} ;

mode \texttt{a} = \texttt{union} ( \texttt{int, real, char} ) ;

mode \texttt{cell} =

struct ( \texttt{real val, int n, ref cell next} )
Ill-formed modes

\[
\text{mode } a = a,
\]

\[
\text{mode } a = b, \quad b = a,
\]

\[
\text{mode } d = \text{struct } (\text{int } n, \text{ d filler})
\]
TALLY :: i; TALLY i.

actual ⟨TALLY1⟩ declarer:
where ⟨TALLY1⟩ is ⟨i⟩, actual declarator;
where ⟨TALLY1⟩ is ⟨TALLY2 i⟩,
    TALLY2 mode indication.

mode a = real, b = a, c = b

TALLY 1 2 3 ... actual-declarer
TALLY 1 2 3 ... mode-indication
Recursive Modes

\[
\text{mode } a = \text{struct} (\text{ref } b \ x, \text{ref } a \ y), \\
b = \text{struct} (\text{ref } a \ x, \text{ref } a \ y)
\]
Recursive modes with finite spelling

```
MODE := PLAIN; STOWED; REF to MODE;
    PROCEDURE; UNITED;
MU definition of MODE; MU application.
MU := mu TALLY.
```

mui definition of structured with reference to
muii definition of structured with reference
to mui application field letter x reference to
mui application field letter y mode field
ter ter x reference to mui application field
letter y mode
Mode spelling

muii definition...

muii definition...

muii application

muii application

muii application

muii application

muii application
Equivalence of modes

WHETHER SAFE1 MOID1 equivalent SAFE2 MOID2:

- where <SAFE1> contains <remember MOID1 MOID2>
  or <SAFE2> contains <remember MOID2 MOID1>,
  WHETHER true;

- unless <SAFE1> contains <remember MOID1 MOID2>
  or <SAFE2> contains <remember MOID2 MOID1>,
  WHETHER <HEAD3> is <HEAD4>
  and remember MOID1 MOID2 SAFE3 TAILETY3 equivalent SAFE4 TAILETY4,
  where SAFE3 HEAD3 TAILETY3 develops from SAFE1 MOID1
  and SAFE4 HEAD4 TAILETY4 develops from SAFE2 MOID2.
develops from

WHETHER SAFE2 HEAD TAILETY
develops from SAFE1 MOID:
- where <MOID> is <HEAD TAILETY>,
  WHETHER <HEAD> shields SAFE1 to SAFE2;
- where <MOID> is <MU definition of MODE>,
  unless <SAFE1> contains <MU has>,
  WHETHER SAFE2 HEAD TAILETY
develops from MU has MODE SAFE1 MODE;
- where <MOID> is <MU application>
  and <SAFE1> is <NOTION MU has MODE SAFE3>
  and <NOTION> contains <yin>
  and <NOTION> contains <yang>,
  WHETHER SAFE2 HEAD TAILETY
develops from SAFE1 MODE.