FORM in CompHEP

Slava Bunichev, Moscow State University

in collaboration with A.Kryukov
CompHEP

• **CompHEP** is a package for computation of Feynman diagram in quantum field theory from Lagrangian until event flow.

• Main feature of the package is an exact symbolic calculation of tree level diagrams for **Standard Model** and beyond (**2HDM**, **MSSM**, **mSUGRA**, ...).

• **CompHEP** is able to compute processes with many particles (up to 6) in the finale state taking into account all the **QCD** and **EW** diagrams, **masses** of fermions and bosons and **width** of unstable particles.

• **CompHEP** has user-friendly **GUI** and highly optimized for fast calculations and effective memory usage.
Price for this approach was high specialization of the module for the symbolic calculations. For example, a fixed number of symbolic structures allows in symbolic expressions under evaluation.

Such specialization is too heavy for further development of CompHEP package!
CompHEP structure:

The **symbolic** part includes diagram generator and symbolic calculator. The **numerical** part includes Monte Carlo integration module and event generator.

We have to change the module for **symbolic** calculations.
To have more advantage symbolic part of CompHEP we incorporate the computer algebra FORM for automatic evaluation of squared Feynman diagrams.

We choose the FORM because:

FORM is rather fast and good quality computer algebra program,
FORM is a standard de-facto in HEP applications.
To save the compatibility of old version of CompHEP with new one and to minimize programming we decide to incorporate FORM in a parallel.

Thus, the new scheme was realized by adding the following modules:

“form_code.c” is the FORM code generator
“procedur.prc” is a set of auxiliary FORM programs
Scheme of the **CompHEP-FORM-CompHEP** interface
Example of diagrams generating

\[ e^+, e^- \rightarrow \mu^+, -\mu^- \]
Example of symbolic calculations
Example of numerical calculations
Example of the **FORM** output

```c
#include<math.h>
define real double
#include"out_ext.h"
#include"out_int.h"
#include"var_def.h"
FNN F1;
real F1(void)
{
    real FACTOR,RNUM,DENOM,result;
    FACTOR= 1./4.*pow(EE,4);

    RNUM= (32*pow(Mm,2) + 32*dp(0) - 64*dp(1)))*dp(0) + 64*pow(dp(1),2);

    DENOM= 4*pow(dp(0),2);
    result=FACTOR*RNUM/DENOM;
    return result;
}
```
We have to optimize C-code for improving calculation time in numerical part.

So we introduce the algorithm for reducing number of multiplications in algebraic expressions.

(results of working this algorithm is similar with Gorner scheme)
Bu-algorithm

$$3d_1d_2d_3 + 5d_2^2d_3 + 7d_2d_3^2 + 2d_1d_3 + 3d_2 =$$

rename: $$d_1 = d_{11}, \ d_1^2 = d_{11}d_{12}, \ ...$$

$$3d_{11}d_{21}d_{31} + 5d_{21}d_{22}d_{31} + 7d_{21}d_{31}d_{32} + 2d_{11}d_{31} + 3d_{21} =$$

find most rare multiplier:

$$3d_{11}d_{21}d_{31} + 5d_{21}d_{22}d_{31} + 7d_{21}d_{31}(d_{32}) + 2d_{11}d_{31} + 3d_{21} =$$

$$3d_{11}d_{21}d_{31} + 5d_{21}(d_{22})d_{31} + 7d_{21}d_{31}(d_{32}) + 2d_{11}d_{31} + 3d_{21} =$$

combine terms with similar terms:

$$3d_{11}d_{21}d_{31} + (5d_{22} + 7d_{32})d_{21}d_{31} + 2d_{11}d_{31} + 3d_{21} =$$

repeat procedure:

$$3(d_{11})d_{21}d_{31} + (5d_{22} + 7d_{32})d_{21}d_{31} + 2(d_{11})d_{31} + 3d_{21} =$$

$$3(d_{11} + 5d_{22} + 7d_{32})d_{21}d_{31} + 2(d_{11})d_{31} + 3d_{21} =$$

$$3(d_{11} + 5d_{22} + 7d_{32})(d_{21})d_{31} + 2(d_{11})d_{31} + 3(d_{21}) =$$

$$((3d_{11} + 5d_{22} + 7d_{32})d_{21} + 2d_{11})d_{31} + 3(d_{21}) =$$

$$((3d_{11} + 5d_{22} + 7d_{32})d_{21} + 2d_{11})d_{31} + 3*3d_{21} =$$

$$((3d_{11} + 5d_{22} + 7d_{32})d_{21} + 2d_{11})d_{31} + 3d_{2}$$
Conclusion

• We completely incorporate FORM in CompHEP

• Crosscheck (we get the same results for both calculators)

• We optimize FORM-output for numerical calculation
Future plans:

- We plan to implement new algorithms to increase efficiency of symbolic calculations.
- Perform calculations in theories with extra dimensions.
- Introduce new complicated structures in the vertices (e.g. Form-factors).
- Use the dimensional regularization.
- Perform polarized calculations by introducing the corresponding density matrices for external lines of squared diagrams.
- Include calculations with extension to 1-loop case.
- Incorporate the gauge invariant classes of diagrams, etc.