The HEPGAME project

J.A.M. Vermaseren

- A little bit of history
- Original targets of the project
- Various failures
- Success or failure?
- Success
- Where does this leave us?

A little bit of history

The HEPGAME project was born out of a sense that strategy in game theory and methodology of computations, in particular solving IBP systems, have much in common. This was already pointed out in 2000 (60-th birthday of F.Yndurain, unpublished) and in Loops and Legs 2004 (”The rules of physics”). It was brought forward (and 1.732 MEuros awarded) in an ERC Advanced grant proposal in 2012 and the project could start on 1-July-2013. Because it is a 5 year project, the money will run out on 30-June-2018. Hence now seems the right moment to look at what has been done and in which direction it takes some of us.

Before the project started I was in the fortunate condition of having Jan Kuipers as a postdoc to help me with Form. He is the one responsible for the initial implementations of polynomial facilities and expression optimization in Form (later upgraded and improved by Takahiro Ueda and Ben Ruijl). With him and the computer scientists Jaap van den Herik and Aske Plaat we designed the first algorithms for the expression simplification and put this as part of the HEPGAME objectives. Some of the people present at this workshop have directly benefitted from this work.

Unfortunately Jan did not want to spend the rest of his life doing big calculations and hence he did not accept my offer to stay a few more years. Not long after the project started he went to Google, but left that company after a few years as well.....

Just a little example what optimization means. Let us assume we have an expression like:

```
F =
   463 + 150*g + 15*g^2 + g^3 - 1152*f - 288*f*g - 18*f*g^2 + 864*f^2 + 108
   *f^2*g - 216*f^3 - 960*e - 240*e*g - 15*e*g^2 + 1440*e*f + 180*e*f*g -
   540*e*f^2 + 600*e^2 + 75*e^2*g - 450*e^2*f - 125*e^3 + 852*d + 228*d*g
    + 12*d*g^2 - 1152*d*f - 144*d*f*g + 432*d*f^2 - 960*d*e - 120*d*e*g +
   720*d*e*f + 300*d*e^2 + 348*d^2 + 48*d^2*g - 288*d^2*f - 240*d^2*e + 64*
   d^3 + 576*c + 144*c*g + 9*c*g^2 - 864*c*f - 108*c*f*g + 324*c*f^2 - 720*
   c*e - 90*c*e*g + 540*c*e*f + 225*c*e^2 + 576*c*d + 72*c*d*g - 432*c*d*f
    - 360*c*d*e + 144*c*d^2 + 216*c^2 + 27*c^2*g - 162*c^2*f - 135*c^2*e +
   108*c^2*d + 27*c^3 + 356*b + 84*b*g + 6*b*g^2 - 576*b*f - 72*b*f*g + 216
   *b*f^2 - 480*b*e - 60*b*e*g + 360*b*e*f + 150*b*e^2 + 408*b*d + 48*b*d*g
    - 288*b*d*f - 240*b*d*e + 96*b*d^2 + 288*b*c + 36*b*c*g - 216*b*c*f -
   180*b*c*e + 144*b*c*d + 54*b*c^2 + 92*b^2 + 12*b^2*g - 72*b^2*f - 60*b^2
   *e + 48*b^2*d + 36*b^2*c + 8*b^3 + 206*a + 54*a*g + 3*a*g^2 - 288*a*f -
   36*a*f*g + 108*a*f^2 - 240*a*e - 30*a*e*g + 180*a*e*f + 75*a*e^2 + 180*a
   *d + 24*a*d*g - 144*a*d*f - 120*a*d*e + 48*a*d^2 + 144*a*c + 18*a*c*g -
   108*a*c*f - 90*a*c*e + 72*a*c*d + 27*a*c^2 + 100*a*b + 12*a*b*g - 72*a*b
   *f - 60*a*b*e + 48*a*b*d + 36*a*b*c + 12*a*b^2 + 23*a^2 + 3*a^2*g - 18*
   a^2*f - 15*a^2*e + 12*a^2*d + 9*a^2*c + 6*a^2*b + a^3;
```

and we have to evaluate it many times as part of a Monte Carlo integration, then we would like to write it such that it has as few arithmetic operations as possible. Hence:

```
Format O3,stats=ON;
Print +f;
.end

  Z1_=3*g;
  Z2_=18 + g;
  Z2_=Z2_*Z1_;
  Z3_=12*d;
  Z4_=4*d;
  Z5_=Z4_ + 2*g;
  Z6_=15 + Z5_;
  Z6_=Z6_*Z3_;
  Z7_=Z1_ + Z3_;
  Z8_=3*b + 25 + Z7_;
  Z9_=4*b;
  Z8_=Z8_*Z9_;
  Z7_=Z7_ + 23;
  Z10_=a + 6*b + Z7_;
```

```
Z10_=a*Z10_;
Z2_=Z10_ + Z8_ + Z6_ + 206 + Z2_;
Z2_=a*Z2_;
Z6_=14 + g;
Z1_=Z6_*Z1_;
Z5_=17 + Z5_;
Z3_=Z5_*Z3_;
Z5_=2*b;
Z6_=Z5_ + Z7_;
Z6_=Z6_*Z5_;
Z1_=Z6_ + Z3_ + 178 + Z1_;
Z1_=Z1_*Z5_;
Z3_=g + 8;
Z6_=Z3_ + Z4_;
Z5_=Z6_ + Z5_;
Z7_=a + 2*Z5_;
Z7_=Z7_*a;
Z3_=Z3_ + 2*d;
```

```
Z8_=8*d;
Z3_=Z3_*Z8_;
Z6_=Z6_ + b;
Z6_=Z6_*Z9_;
Z8_=g + 16;
Z8_=Z8_*g;
Z3_=Z7_ + Z6_ + Z3_ + Z8_ + 64;
Z5_=a + Z5_;
Z6_=c + Z5_;
Z7_=3*c;
Z6_=Z6_*Z7_;
Z6_=Z6_ + Z3_;
Z6_=c*Z6_;
Z8_=Z7_ + 2*Z5_;
Z8_=Z8_*Z7_;
Z3_=Z3_ + Z8_;
Z5_=Z5_ + Z7_;
Z7_=5*e;
```

```
Z8_=3*Z5_ - Z7_;
Z8_=Z8_*Z7_;
Z8_= - 3*Z3_ + Z8_;
Z8_=Z8_*Z7_;
Z9_=2*Z5_ - Z7_;
Z9_=Z9_*Z7_;
Z5_= - 2*f - Z7_ + Z5_;
Z5_=f*Z5_;
Z3_=6*Z5_ + Z9_ - Z3_;
Z3_=f*Z3_;
Z5_=19 + g;
Z5_=g*Z5_;
Z5_=71 + Z5_;
Z7_=29 + 4*g;
Z7_=3*Z7_ + 16*d;
Z7_=d*Z7_;
Z5_=3*Z5_ + Z7_;
Z4_=Z5_*Z4_;
```

```
      Z5_=15 + g;
      Z5_=g*Z5_;
      Z5_=150 + Z5_;
      Z5_=g*Z5_;
      F=18*Z3_ + Z8_ + 9*Z6_ + Z2_ + Z1_ + Z4_ + 463 + Z5_;
```

```
*** STATS: original 7P 299M 119A : 432
*** STATS: optimized 0P 51M 57A : 108
```

and now, instead of 313 multiplications and 119 additions/subtractions we have 51 multiplications and 57 additions/subtractions.

Original targets

The objectives of the proposal were

1. : To create a system that can evaluate expressions in as few numerical operations as possible (in the practical, not in the mathematical sense).
2. : Use/implement this in Form and apply it to the Grace system.
3. : Automatically solve IBP systems parametrically.
4. : Use the above for 3-loop DIS polarized scattering.
5. : Try to solve the 4-loop mass anomalous dimension of quarks.
6. : Holy grail: make a 4-loop version of Mincer and use it.
7. : Study parallelization and see what can be applied to make 1 and 2 faster. In particular try to use the Intel Phi board.

Note that this was the wish-list in 2012.

We got awarded two graduate students and 10 postdoc years. In addition there was money for some computers, travel and visitors. The graduate students were supposed to work more on the computer aspects, while one postdoc was supposed to be in a field between calculations and Form and the others more in the physics of phenomenological calculations. The graduate students would be based with the AI people, originally in Tilburg, but later, when the AI group moved to Leiden, they also went to Leiden. Officially their thesis advisor would be Jaap van den Herik, a well known person in game theory.

Because the timing of these grants is rather out-of-phase with the HEP postdoc market, I managed to hire only one postdoc who could start in the autumn of 2013: Matthias Ritzmann. We did manage to hire two graduate students: Ben Ruijl and Ali Mirsoleimani. A year later (2014) I could hire Takahiro Ueda and Franz Herzog, and after Matthias left (2015), we found Giulio Falcioni willing to join the group (2016). Takahiro left recently for a position in Japan. Franz got a prestigious Dutch grant (2017) and is still at Nikhef. Ben got his PhD last autumn and is now a postdoc at the ETH, while Ali will get his PhD this autumn, but works already at ASML, a world leader in the production of machines that make computer chips. Giulio will go to Edinburg this autumn. In total we used the two graduate positions fully and spent 10.5 years of postdoc money.

In addition collaborations continued with Andreas Vogt (Liverpool) and Sven Moch (DESY-Zeuthen and later Hamburg) and occasionally we had visitors.

Various failures

What happened?

To end on a positive note, we will look at the failures first.

Parallelization project

The task of Ali Mirsoleimani was to study to what extend we would be able to improve parallelization in our calculations. For this one of the computers we bought was equiped with an Intel Phi board. He spent a lot of time studying this and made some very nice measurements of its performance which were presented at conferences (the major way of presenting results in this field), but the eventual conclusion was that unfortunately it was not going to help in physics calculations based on computer algebra. I am not talking about for instance Monte Carlo integration and that type of calculations, but only about symbolic manipulations. The bottleneck of data transfer rates more than offset the increased CPU power. Recently I remarked to one of our systems managers that we should have used the Phi to mine bitcoins, but he told me it was also not very good at that......

After this Ali has been studying Monte Carlo Tree Search methods in order to make them parallel without incurring the traditional penalties. In this he has been quite succesful, but his contract expired before he could implement this in the Form optimization routines.

Quark mass anomalous dimension

When Matthias Ritzmann came to Nikhef he was still heavily involved with the construction of the program VINCIA. Of course one does not say: drop it. It took however much longer than expected. The VINCIA effort, although not part of the HEPGAME project, was at last succesful. At the same time he did start looking at the quark mass anomalous dimension. This is not an easy problem as some people here can tell us all about. When Franz came to Nikhef he went to work with Matthias on the problem, but most of the fun went out of it when the Karlsruhe group (Marquard, Smirnov, Smirnov and Steinhauser) came with a mostly algebraic but also partially numerical solution. In the end Matthias decided to leave the field, also because he really wanted to go back to Switserland.

Success or failure?

Three loop polarized DIS

And then a project that is somewhere between a success and a failure: the three loop polarized DIS scattering. From the physics viewpoint it is a big success because we solved it. From the viewpoint of AI it is a failure because we found a shortcut and hence did not need the AI solution we had hoped to find for it (we did it with RI). How?

When you determine moments of structure functions, up to three loops the general formula is a linear combination of harmonic sums of $(N+i)$ with i a small integer. The coefficients seem to be very simple fractions with denominators only containing small prime numbers like 2, 3 and 5. Hence if we calculate for instance an 11-th moment and we see $1/11^5$ there must be some harmonic sum of weight 5. Better yet: if I calculate the moments 9, 11,13, and the factor $1/11^5$ shows up at the 9-th moment for the first time and $1/13^5$ in the 11-th moment then I know that there is an argument $(N+2)$.

By doing a lot of this kind of numerology, and by calculating enough moments by brute force, and by combining this with many physics constraints on the difference between polarized and unpolarized scattering, one can bring this down to only very few unknown coefficients. These were then resolved with LLL techniques. After indeed getting a 'simple' solution, two more moments were computed (N=27 and N=29) and they did fit the formula. Most of this was done by Andreas Vogt.

When we showed this to Sven Moch he made us aware of an earlier paper by Velyzhanin who used a similar technique. Hence we do not claim to be the first to use this method although we do claim independence. More important: it worked! But it took away the need to automate the work that was done by hand for the unpolarized all-N splitting and coefficient functions.

The technique is currently being used for some of the simpler color/flavor channels at the 4-loop level (see talk by Andreas Vogt).

Success

And now the success stories. This list is longer.

Optimization

Ben Ruijl spend much of his first year studying the MCTS method and found an alternative way to implement code optimization in Form via a technique called hill climbing. It turns out that the sample formulas we used, which came from some nasty diagrams produced by the GRACE system, are very suitable for this technique. He has presented this at several computer conferences. He implemented this in Form. This is "Format O4". It gives about the same number of multiplications and additions/subtractions as O3, but in less time.

At the same time we experimented a bit with Grace specific improvements in the Form language. By making systematic changes of variables one can often make formulas shorter, even without increasing the number of variables. Like replacing x_1 by $3-x_1-2\,x_2$. After this one can still use other improvements and the formulas can be optimized even further. The full method was applied in a number of papers by the GRACE group. The Form optimizations like MCTS and hill climbing are in use by other groups as well.

This field of research is still open ended, and the hope was to pursue it further as part of a new ERC advanced grant, but unfortunately this was not awarded. The goal was to have some deep learning AI being able to reconstruct the above expression to

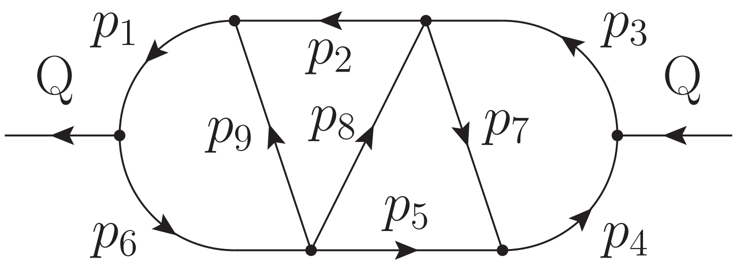
```
(a+2*b+3*c+4*d-5*e-6*f+g+8)^3
-(a-2*b+6*d-3*g-7)^2
```

It should be fairly easy to train such a program, as one can provide an infinite number of examples.

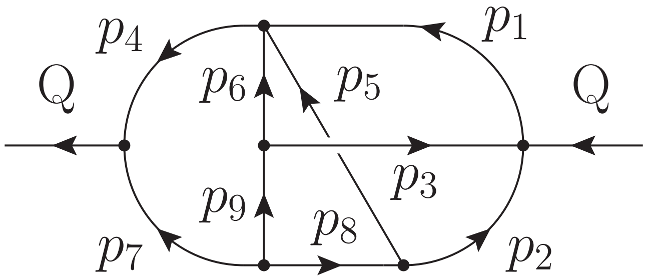
Forcer

A really big success started with a good look at why until then a 4-loop mincer had eluded us. At this time we knew already that it must be possible because Roman Lee had a Mathematica program for it (LiteRed), even though it was unpractically slow. We considered it a proof of existence.

We started with looking at why the Laporta algorithm can be successful. This led to a crossbreed of the parametric and the Laporta methods. Suddenly a diagram that I could never solve before, became moderately simple.



This was in July 2014. Hence Ben and I and a little later also Takahiro, set out to see whether we could indeed do all topologies and create the 4-loop program which we called Forcer. Work proceeded steadily, but there was one topology that was really tough, even though it does not correspond to a master integral.



It was a good collaboration where each of us made vital contributions. At the same time Form needed to be extended a bit to make it faster for certain types of operations. After about a year forcer started working and we could start with calculations. First redoing some older calculations, but soon also getting new results like the 4-loop beta function with all powers of the gauge parameter ξ etc. This done in a background gauge. Etc.

And then we started also looking at the 4-loop moments of structure functions. This is still rather tough, because 4-loop calculations take much more time than 3-loop calculations. At the moment we made it up to N=8 for the non-singlet and N=6 for the singlet. Andreas can tell you more about this.

An improvement came, when in collaboration with Sven Moch we got a generalization of the quark operator that one needs for the splitting functions. This reduces the calculation of moments to much simpler integrals as one starts from three-point functions, rather than four-point functions and hence we can calculate more moments, although now only for splitting functions, not for the coefficient functions. The other operators are still under investigation. There operators that give the ghost and non-gauge invariant contributions are a major headache. Once we have those we can probably obtain enough moments for a meaningful N^3LO analysis.

Into the infrared

The question arose whether it is possible to do the various subtractions of divergences in Feynman diagrams automatically and on a diagram by diagram level. And even if this would be possible, we knew it could not possibly be easy, because Kostja Chetyrkin has been spending a good part of his career pursuing such subtraction methods. According to the book it seemed not impossible, but the book concentrates more on ϕ^4 and hence many vital details are missing. Franz and Ben set out to study this, with as ultimate goal combining it with the Forcer program to obtain the 5-loop beta function for general color factors.

Things went, as usual, a bit slower than hoped for, so at LL2016 Kostja presented the Karlsruhe results (Baikov, Chetyrkin, Kühn) for QCD in a global subtraction method that he developed. We discussed this with Kostja and decided that we would collaborate with him on working out his global method further for a general gauge group (more about this later) but this time using Forcer. At the same time we would continue with the 'local' subtraction method that Franz and Ben were developing. My expectation at the time was that the collaboration with Kostja would first come up with the general formula, and the local result would come later.

It worked out differently. The global method ran into several difficulties and suddenly all the pieces for the local method fell in place and after just a few days of calculations the result was there (1-1-2017) (good way to start the year....), this time for a general gauge group. It led to a few papers. It showed that the method of local Rstar did work and although the Rstar program can still be improved, both in generality and in speed, it has already become a useful tool. More about it you can learn from Ben Ruijl.

At the same time the global Rstar method, which was now worked out by Giulio and Franz, under the guidance of Kostja and with Form and color assistance by me, proceeded slowly, because the operator mixings became extremely complicated. Eventually Giulio had a very bright idea and it was finished after all. To spice up the paper the whole of renormalizations of the vertices was produced. The longer paper about the method is still to be produced though. There are many interesting aspects there, including the part about color factors.

To show some of the complications:

We needed the 3-loop corrections to the 4-gluon vertex, including all its color/Lorenz channels. Producing projection operators for this proved too complicated. I show here a little bit of the mess:

First define

$$
\begin{aligned}
\Delta^{\mu_1\mu_2\mu_3\mu_4} &= \delta^{\mu_1\mu_2}\delta^{\mu_3\mu_4} + \delta^{\mu_1\mu_3}\delta^{\mu_2\mu_4} + \delta^{\mu_1\mu_4}\delta^{\mu_2\mu_3} \\
d_R^{a_1a_2a_3a_4} &= \frac{1}{6}(T^{a_1}_{i_1i_2}T^{a_2}_{i_2i_3}T^{a_3}_{i_3i_4}T^{a_4}_{i_4i_1} + T^{a_1}_{i_1i_2}T^{a_2}_{i_2i_3}T^{a_4}_{i_3i_4}T^{a_3}_{i_4i_1} + T^{a_1}_{i_1i_2}T^{a_3}_{i_2i_3}T^{a_2}_{i_3i_4}T^{a_4}_{i_4i_1} \\
&\quad +T^{a_1}_{i_1i_2}T^{a_3}_{i_2i_3}T^{a_4}_{i_3i_4}T^{a_2}_{i_4i_1} + T^{a_1}_{i_1i_2}T^{a_4}_{i_2i_3}T^{a_2}_{i_3i_4}T^{a_3}_{i_4i_1} + T^{a_1}_{i_1i_2}T^{a_4}_{i_2i_3}T^{a_3}_{i_3i_4}T^{a_2}_{i_4i_1}) \\
d_A^{a_1a_2a_3a_4} &= \frac{1}{6}(f^{j_1j_2a_1}f^{j_2j_3a_2}f^{j_3j_4a_3}f^{j_4j_1a_4} + f^{j_1j_2a_1}f^{j_2j_3a_2}f^{j_3j_4a_4}f^{j_4j_1a_3} \\
&\quad +f^{j_1j_2a_1}f^{j_2j_3a_3}f^{j_3j_4a_2}f^{j_4j_1a_4} + f^{j_1j_2a_1}f^{j_2j_3a_3}f^{j_3j_4a_4}f^{j_4j_1a_2} \\
&\quad +f^{j_1j_2a_1}f^{j_2j_3a_4}f^{j_3j_4a_2}f^{j_4j_1a_3} + f^{j_1j_2a_1}f^{j_2j_3a_4}f^{j_3j_4a_3}f^{j_4j_1a_2})
\end{aligned}
$$

These are totally symmetric objects. If we define f and T to have color dimension 1/2 the d_R and d_A tensors have color dimension 2. The lowest order vertex has color dimension 1. At L loops in perturbation theory we need color/Lorentz invariants for which the whole formula is symmetric between the gluons and have color dimension L+1. Hence

$$
\begin{aligned}
0Loops &\rightarrow V^{(0)} \\
1Loop &\rightarrow d_A^{a_1a_2a_3a_4}\Delta^{\mu_1\mu_2\mu_3\mu_4} \\
&\rightarrow d_R^{a_1a_2a_3a_4}\Delta^{\mu_1\mu_2\mu_3\mu_4} \\
2Loops &\rightarrow d_A^{a_1a_2j_1j_2}f^{a_3j_1j_3}f^{a_4j_2j_3}\delta^{\mu_1\mu_2}\delta^{\mu_3\mu_4} + d_A^{a_1a_3j_1j_2}f^{a_2j_1j_3}f^{a_4j_2j_3}\delta^{\mu_1\mu_3}\delta^{\mu_2\mu_4} \\
&\quad +d_A^{a_1a_4j_1j_2}f^{a_2j_1j_3}f^{a_3j_2j_3}\delta^{\mu_1\mu_4}\delta^{\mu_2\mu_3} \\
&\rightarrow d_R^{a_1a_2j_1j_2}f^{a_3j_1j_3}f^{a_4j_2j_3}\delta^{\mu_1\mu_2}\delta^{\mu_3\mu_4} + d_R^{a_1a_3j_1j_2}f^{a_2j_1j_3}f^{a_4j_2j_3}\delta^{\mu_1\mu_3}\delta^{\mu_2\mu_4} \\
&\quad +d_R^{a_1a_4j_1j_2}f^{a_2j_1j_3}f^{a_3j_2j_3}\delta^{\mu_1\mu_4}\delta^{\mu_2\mu_3} \\
3Loops &\rightarrow (d_A^{a_1a_2j_1j_2}d_A^{a_3a_4j_1j_2} + d_A^{a_1a_3j_1j_2}d_A^{a_2a_4j_1j_2} + d_A^{a_1a_4j_1j_2}d_A^{a_2a_3j_1j_2})\Delta^{\mu_1\mu_2\mu_3\mu_4} \\
&\rightarrow (d_R^{a_1a_2j_1j_2}d_A^{a_3a_4j_1j_2} + d_R^{a_1a_3j_1j_2}d_A^{a_2a_4j_1j_2} + d_R^{a_1a_4j_1j_2}d_A^{a_2a_3j_1j_2})\Delta^{\mu_1\mu_2\mu_3\mu_4}
\end{aligned}
$$

Unfortunately the color program has a problem with open gluon indices and also the number of color dimensions would become too large for projection operators. Hence we had to modify the color program a bit and then work out the various channels. This gave way too many channels, and a very creative way of applying Jacobi identities was needed to bring this to a minimal form. An example:

$$\begin{aligned} d_R^{a_1a_4j_1j_2}\ f^{a_2j_1j_3}\ f^{a_3j_2j_4} &= +\, d_R^{a_1a_2a_3a_4}\ c_a \\ &\quad -\, d_R^{a_1a_2j_1j_2}\ f^{a_3j_1j_3}\ f^{a_4j_2j_3} \\ &\quad -\, d_R^{a_1a_3j_1j_2}\ f^{a_2j_1j_3}\ f^{a_4j_2j_3} \end{aligned}$$

This leads eventually to a full 3-loop result with all powers in the gauge parameter of which here a little piece without gauge parameter:

```
Time =     162.24 sec    Generated terms =        113
              F1         Terms in output =        113
                         Bytes used      =       4232

Time =     162.24 sec    Generated terms =        305
              F2         Terms in output =        305
                         Bytes used      =      11292

Time =     162.24 sec    Generated terms =       1065
              F3         Terms in output =       1065
                         Bytes used      =      39668

   F1=
      +V4g(a1,a2,a3,a4,mu1,mu2,mu3,mu4)*(2/3*ep^-1*nf*[1]-1/8*ep^-1*ca*[6]+29/
         24*ep^-1*ca*[3]-5/3*nf*[1]+17/48*ca*[6]-127/48*ca*[3]+1/2*ep*nf*[1]-31/
         96*ep*ca*[6]+11/32*ep*ca*[3]+2/3*ep*z2*nf*[1]-1/8*ep*z2*ca*[6]+29/24*
         ep*z2*ca*[3]+1/4*ep^2*nf*[1]+11/64*ep^2*ca*[6]+11/64*ep^2*ca*[3]+4/3*
         ep^2*z3*nf*[1]-1/4*ep^2*z3*ca*[6]+29/12*ep^2*z3*ca*[3]-5/3*ep^2*z2*nf*
         [1]+17/48*ep^2*z2*ca*[6]-127/48*ep^2*z2*ca*[3]+1/8*ep^3*nf*[1]+11/128*
         ep^3*ca*[6]+11/128*ep^3*ca*[3]+19/6*ep^3*z4*nf*[1]-19/32*ep^3*z4*ca*
         [6]+551/96*ep^3*z4*ca*[3]-10/3*ep^3*z3*nf*[1]+17/24*ep^3*z3*ca*[6]-127/
         24*ep^3*z3*ca*[3]+1/2*ep^3*z2*nf*[1]-31/96*ep^3*z2*ca*[6]+11/32*ep^3*
         z2*ca*[3]+1/16*ep^4*nf*[1]+11/256*ep^4*ca*[6]+11/256*ep^4*ca*[3]+4*
         ep^4*z5*nf*[1]-3/4*ep^4*z5*ca*[6]+29/4*ep^4*z5*ca*[3]-95/12*ep^4*z4*nf
      *[1]+323/192*ep^4*z4*ca*[6]-2413/192*ep^4*z4*ca*[3]+ep^4*z3*nf*[1]-31/
      48*ep^4*z3*ca*[6]+11/16*ep^4*z3*ca*[3]+1/4*ep^4*z2*nf*[1]+11/64*ep^4*
      z2*ca*[6]+11/64*ep^4*z2*ca*[3]+4/3*ep^4*z2*z3*nf*[1]-1/4*ep^4*z2*z3*ca
      *[6]+29/12*ep^4*z2*z3*ca*[3])
   +[dA](a1,a2,a3,a4)*dd(mu1,mu2,mu3,mu4)*(3/2*ep^-1*[6]-7/4*ep^-1*[3]+1/4*
      ep^-1*[2]-1/4*[6]+29/24*[3]-7/24*[2]-49/8*ep*[6]+619/144*ep*[3]-41/144
      *ep*[2]+3/2*ep*z2*[6]-7/4*ep*z2*[3]+1/4*ep*z2*[2]+15/16*ep^2*[6]+617/
      864*ep^2*[3]-163/864*ep^2*[2]+3*ep^2*z3*[6]-7/2*ep^2*z3*[3]+1/2*ep^2*
      z3*[2]-1/4*ep^2*z2*[6]+29/24*ep^2*z2*[3]-7/24*ep^2*z2*[2]+15/32*ep^3*
      [6]-629/5184*ep^3*[3]-569/5184*ep^3*[2]+57/8*ep^3*z4*[6]-133/16*ep^3*
      z4*[3]+19/16*ep^3*z4*[2]-1/2*ep^3*z3*[6]+29/12*ep^3*z3*[3]-7/12*ep^3*
      z3*[2]-49/8*ep^3*z2*[6]+619/144*ep^3*z2*[3]-41/144*ep^3*z2*[2]+15/64*
      ep^4*[6]-6847/31104*ep^4*[3]-1867/31104*ep^4*[2]+9*ep^4*z5*[6]-21/2*
      ep^4*z5*[3]+3/2*ep^4*z5*[2]-19/16*ep^4*z4*[6]+551/96*ep^4*z4*[3]-133/
      96*ep^4*z4*[2]-49/4*ep^4*z3*[6]+619/72*ep^4*z3*[3]-41/72*ep^4*z3*[2]+
      15/16*ep^4*z2*[6]+617/864*ep^4*z2*[3]-163/864*ep^4*z2*[2]+3*ep^4*z2*z3
      *[6]-7/2*ep^4*z2*z3*[3]+1/2*ep^4*z2*z3*[2])
   +[dR](a1,a2,a3,a4)*dd(mu1,mu2,mu3,mu4)*(-4/3*[1]+26/9*ep*[1]-1/27*ep^2*
      [1]-4/3*ep^2*z2*[1]-83/162*ep^3*[1]-8/3*ep^3*z3*[1]+26/9*ep^3*z2*[1]-
      409/972*ep^4*[1]-19/3*ep^4*z4*[1]+52/9*ep^4*z3*[1]-1/27*ep^4*z2*[1]);
 +  .....
```

Because we need to study all operator mixings the original external vertex needs to be marked. This is what the `[1]`, `[2]`, `[3]`, `[6]` are for.

To compare methods:

In terms of computer resources the global Rstar is fastest, but it is conceptually the most complicated and for now it needs most human intervention. But we could compute the 5-loop beta function for general gauge group including a power of the gauge parameter which no other method has been capable of doing.

Second is the local Rstar. It is much simpler conceptually but slower in execution. In terms of computer resources it needed about the same amount of CPU power as the global Rstar but the local method was without a gauge parameter.

From the viewpoint of physics the simplest method may be the regularization of all IR divergences with a uniform mass as done by Luthe, Maier, Marquard and Schröder, but it takes much computer time and also the development of the programs for the massive vacuum bubble diagrams is far from trivial.

What is in a sense truely amazing is that all these calculations gave independently the same answer. With independently I mean that none was working towards an existing result, but all came to an answer by internal methodology.

With the local Rstar operation it is also possible to compute a few more 5-loop quantities like Higgs decay and moments of splitting functions. The latter is very slow though and requires very much computer time. At the moment we have the N=2 and N=3 moments of the nonsinglet splitting functions.

Currently Giulio and Franz are studying whether it is possible to combine the operator method for splitting functions with the global Rstar of Kostja, which is much faster than the local Rstar. One might obtain more moments of splitting functions this way. The jury is still out on that one.

As you can see, we have been rather busy and although we were not the only ones working in the field of extreme calculations, and even though we were several times not even first, we managed to produce a few nice and sometimes even unique results and programs. It was proposed to pursue the derivation of parameteric reduction identities by means of AI optimization techniques (we never really got around to that, because the Forcer-Rstar path was going so well), but alas, as mentioned the new proposal was not accepted. Hence other people will have to do that.

Where does this leave us?

The diaspora is in full swing now.

We still have a few computers that are really good for big Form programs. The nicest one is a Fujitsu machine with 32 cores, 756 Gbytes of memory and 20 Tbytes of good quality SSD (in addition to 6 Tbytes of regular disks).

Currently Ben wants to set up Form for the methods of the future that will allow the use of much more AI. The first step is to reprogram Form to a more modern language (Rust) in such a way that he and Takahiro control every aspect of it.

I can only applaud this. I am even helping them, but it is their project.