



ELSEVIER

Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

Nuclear Instruments and Methods in Physics Research A 534 (2004) 284–288

NUCLEAR
INSTRUMENTS
& METHODS
IN PHYSICS
RESEARCH
Section A

www.elsevier.com/locate/nima

DIANA and applications to fermion production in electron–positron annihilation [☆]

J. Fleischer^{a,*}, A. Lorca^{b,2}, M. Tentyukov^c

^a*Fakultät für Physik, Universität Bielefeld, Institut für Theoretische Physik, Universitätsstrasse 25, D-33615 Bielefeld, Germany*

^b*DESY Zeuthen, Platanenallee 6, D-15738 Zeuthen, Germany*

^c*Institut für Theoretische Teilchenphysik, Universität Karlsruhe, D-76131 Karlsruhe, Germany*

Available online 3 August 2004

Abstract

A demonstration of the use of DIANA has been given. As an example, the process $e^+e^- \rightarrow t\bar{t}$ was selected. Recent developments are indicated: automation of momenta distribution, the status of parallelization of FORM jobs, handling of topologies and new features for the implementation of fermions.

© 2004 Elsevier B.V. All rights reserved.

1. Introduction

The C-program DIANA (DIagram ANALyser) for the automatic Feynman diagram evaluation was first documented in Ref. [1].³ Meanwhile, there exists a series of applications [2], which were performed with the help of DIANA. In this contribution, we concentrate on the more technical

aspects of fermion production and DIANA while physics can be taken from these papers.

2. Automation of momenta distribution

Starting from Version 2.25, momenta can be introduced automatically, all momenta being balanced in the vertices. Optionally, the user can point out which lines should carry bare integration momenta. All remaining momenta will be assigned by DIANA [3].

Sometimes it is necessary to use more sophisticated momenta distributions: e.g. momenta like $k - p_1, k - p_2, k - p_3$, etc. assigned to some definite lines (see Ref. [4]). For such cases, DIANA provides the possibility to define momenta only for the virtual lines, attaching the external legs [3], i.e. some topologies may occur with different external lines.

[☆]Work supported in part by DFG-Sonderforschungsbereich/Transregio 9 ‘Computergestützte Theoretische Teilchenphysik’.

*Corresponding author.

E-mail address: fleischer@physik.uni-bielefeld.de

(J. Fleischer).

¹Thanks to DESY Zeuthen for various invitations.

²Supported in part by European Community’s Human Potential Programme under contract HPRN-CT-2000-00149.

³See also <http://www.physik.uni-bielefeld.de/tentyukov/diana.html>

In version 2.29, this approach was extended to systematic and automatic distribution of “chords” (c_i) and “bridges” (b_i) which can be expressed in terms of external momenta, see Fig. 1. In the demo below (2ttbar) this is demonstrated.

Sometimes, topologies are generated from more complicated ones by scratching lines. In such cases, one wants to stick to the momenta introduced for the lines which are kept [3].

A more difficult situation appears when the user wants to “scratch” some external lines. Consider, e.g. the Anomalous Magnetic Moment; this is a 3-point function, fermion–fermion with an external photon of zero momentum. After differentiating some diagram and putting the photon momentum to zero, the resulting graph appears to be a 2-point function with higher degree of propagators. So we need to map the momenta of the 3-point function to the corresponding momenta of a 2-point function, putting one external momentum to zero.

Starting from version 2.35, DIANA is able to map momenta from n -point functions to m -point functions ($m < n$).

3. Demonstration

The purpose of our demonstration is to support for potential users an easy access to DIANA. The

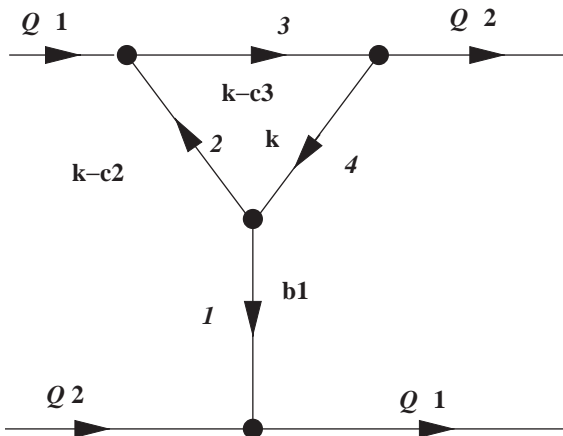


Fig. 1. Here k is the integration momentum, $b_1 = Q_1 - Q_2$, $c_2 = Q_1 - Q_2$, $c_3 = -Q_2$.

directory `init`⁴ is provided and, having installed DIANA on his computer, the user can copy it (under UNIX: `cp -r init demo`) to his own directory `demo`. There he will find three subdirectories (`1ttbar`, `2ttbar`, `3ttbar`) with different status of DIANA running. Following the instructions below, he can easily learn how to use DIANA.

1ttbar

- No topology editor (`tedi`) at generation time—the topologies are already ‘nice’—edit `topol.tt` or run `tedi topol.tt`; ‘create’ file is available.
- call: `diana35 -c create.tt`; `config.tt` is produced.
- run: `diana35 -c config.tt`
- the ‘FORM-input’ is found in ‘`tt.in`’; `tt.ps` and `ttInfo.ps` contain the pictures of all and single diagrams, respectively.
- edit `config.tt` and uncomment `SET MakeInfoEps = “!”` and `SET MakeEps = “!”` and run again: the result will be collected in the directories `EPS` and `InfoEPS`; these contain encapsulated `ps` files for all diagrams—ready to use in a publication.

2ttbar

- The topologies are not yet prepared: call `diana35 -c create.tt` and `tedi` will be invoked. The shapes of the diagrams are ugly. The user can change the shapes by clicking the vertices, move the cursor and also by some automatism clicking the icons on the upper line.
- momenta are produced automatically with conservation in the vertices.
- edit `create.tt` and uncomment `SET _NO_TABLES=NO` and `SET _NO_TABLE_MOMENTA=NO` and run `diana35 -c create.tt` again; `config.tt` is produced.
- run: `diana35 -c config.tt`; the produced ‘FORM-input’ is found in ‘`tt.in`’
- in ‘`tt.in`’ all momenta in Feynman integrands are expressed in terms of “bridges” $b\#$ and “chords” $c\#$ (due to the lines `\token(bridge,b)` and `\token(chord,c)` in ‘`create.tt`’).

⁴Can be obtained via WWW: <http://www.physik.uni-bielefeld.de/tentukov/acat03.html>

Additionally, DIANA produced defines expressing bridges in terms of chords, and chords in terms of external momenta.

3ttbar

- Here, fixing of the bare integration momenta is explained (the shapes are nice): in create.tt SET_MARK_LOOP = YES is uncommented.
- call: diana35 -c create.tt; tedi will be invoked with the prompt Arrange_loop_momenta.
- go to any topology and put marks with the right mouse button where you want the bare integration momentum: clicking it you set–unset the loop mark. After exiting tedi, it will be invoked by DIANA again with the resulting momenta distribution. Check the result.

4. Named, sorted and reshaped topologies

In many cases, it will be helpful to load some predefined topologies for simple processes containing up to 1-loop or 5-legs and the full set of appearing topologies being already prepared (for 2legs1loops, {3,4}legs{0,1}loops and 5legs0loops) with a visual aspect nicer than the default construction, a given name related with legs-channel-degeneracy-looptype information and fixed momenta distribution obeying symmetry properties. The arrangement of momenta follows a clockwise indexing with notation for external momenta p_i , integration loop-momenta q and internal momenta decomposition k_i (e.g. Fig. 2).

All the topologies can be found in the tml folder, accessible as in the following example for $2 \rightarrow 2$ process and 1-loop.

```
\Begin(topology,4legs1loops.top)
```

5. Algorithm for topology clustering

The idea of topology clustering offers the possibility to organize the later processing of diagrams according to certain topology classes. As soon as the diagrams are invoked in a successive

Topology 4s2c:

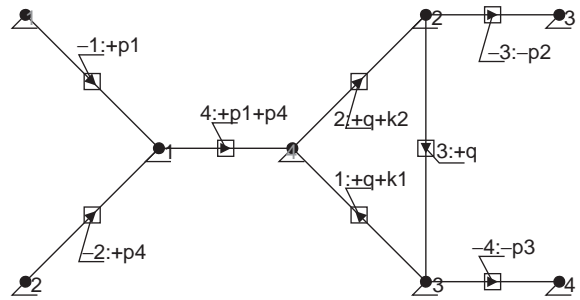


Fig. 2. Example of momenta distribution in topology 4s2c.

order of topologies it can be recognized which is the first and the last diagram within the same class.

For example, if our intention is to further calculate with FORM3 [5], such information is then stored into initialization values for dollar variables (\$), allowing a flexible use in #do loops. Something like this is written:

```
#define topologiesnames
"4s2c,4s4b,4s3b"
#$first4s3b=91;
#$last4s3b=114;
#$first4s4b=67;
#$last4s4b=90;
#$first4s2c=1;
#$last4s2c=66;
```

6. Fermion current analysis

Since much information related to external fermions can be extracted generically for all the diagrams, some functions were developed to decide whether a diagram belongs to the fermion channel "S", "T" or "U" without analyzing the topology itself, just comparing the indices given to external fermion lines. Also, new variables are set, telling us to which fermion line-index belongs any external particle and giving a suitable definition of external momenta required in the evaluation of matrix elements. This feature is particularly useful when several channels are in connection as shown by Ref. [6] for Bhabha scattering.

7. Fermion color implementation

Since many of the calculations carried out with DIANA were related to electroweak processes, it has been fruitful to simplify the models without color indices for quarks. Nevertheless, when tracing fermion loops or summing over final states, it becomes mandatory to check which color the fermions carry in the process. By telling DIANA which of the prototypes in our model file are with `color=3` (`u,d` are the prototypes for our quarks in the model file), a pair of variables is defined for each diagram concerning internal fermion loops and the color is added to each of the external fermions. For instance, a quark loop in a self-energy of any propagator mediating lepton–lepton scattering will lead to:

```
# define FERMIONLOOP "1"
# define COLORFACTOR "3"
...
# define color1 "1"
# define color2 "1"
# define color3 "1"
# define color4 "1"
```

8. Parallel computation of Feynman diagrams with DIANA

After the diagrams are generated, DIANA may be used as a “control center” for further evaluations (in parallel) by invoking the corresponding programs and providing them the generated input. To avoid a cluster/processor overloading, each time only one job per node/processor is actually running while all the rest are queued. Such an approach is known as a Batch Queueing System (BQS).

There are two kinds of optimization of BQS available. The first is for the case when the average number of jobs is much larger than the number of nodes. In this case, BQS usually provides various complicated methods for scheduling, message pasting, job run-time quotas and resource management in order to distribute computational resources among various tasks of all kinds. In our case, all these mechanisms are not needed. Secondly, if the average number of jobs is

comparable with the number of nodes, which is typical for parallel computation, a BQS should provide load balancing, process migration mechanisms and some others.

Compared to that, DIANA has to implement parallelism in terms of job queueing, and the typical situation consists of a long queue of nearly even tasks with the same priority. Evidently, we need not any load balancing. Indeed, the best procedure is to send a new task to a node immediately when the node becomes free. Thus, the real problem in our case is the problem of synchronization. Traditionally, this problem is ignored in BQS since all jobs are assumed to be completely independent. The synchronization problem arises in true parallel systems. Since we use queueing in order to implement a parallelism, the problem is of immediate interest in our case.

The implementation is based on an extension of DIANA’s built-in language [7], called “TM - language” (text manipulating). Similar to the TeX language, each command has to begin with an escape (“\”) character, while all lines without escape—characters are simply typed to the output file.

As an example, let us consider the following script `runf`, which is used to run FORM jobs in parallel after the FORM input has been produced in a folder:

```
#!diana -smp 1 -c runpar.tml
\STARTSERVERS (phy25, phy26, phy27,
phy28)
\system(echo > log.all)

\REPEAT(N)
\exec(form -d i=\get(N) do.frm > /tmp/
log.\get(N))
\stick(cat /tmp/log.\get(N) >>
log.all)
\stick(rm /tmp/log.\get(N))
\ENDREPEAT()

\wait(2000)
```

The user enters: `runf 186 200`, and the system executes:

```
diana -smp 1 -c runpar.tml runf 186 200.
```

The file `runpar.tml` contains definitions of various TM functions and some settings.

The script is more or less self-explaining. The interested user may look at [7] for details. The script will try to initialize DIANA servers on computers, e.g., `phya25,phya26,phya27` and `phya28`, and then execute all diagrams in parallel on all these computers.

All the instructions between `\REPEAT(N)` ... `\ENDREPEAT()` are cycled with `N=186,...,200`. We assume that there is some folder file, say, `tt.in` with FORM input for each diagram. The FORM program `do.frm` evaluates a diagram by virtue of including a fold from the folder `tt.in` via an instruction like `#include tt.in # n'i'`. The macro definition `i` comes from the command line `form -d i=\get(N) do.frm`, where `\get(N)` runs from 186 to 200. Each FORM job saves the result to the local directory, but the corresponding concatenation is performed by `\stick(cat ...)` on the same computer. At the end, all results will be collected in the file `log.all`, and all intermediate files `\tmp\log.#` will be removed.

After all jobs are queued, the function `\waitall(2000)` will report every 2 s how many jobs are not yet completed.

There is the possibility to build a really powerful BQS based on these TM extensions.

References

- [1] M. Tentyukov, J. Fleischer, *Comput. Phys. Commun.* 132 (2000) 124.
- [2] J. Fleischer, et al., *Phys. Lett. B* 459 (1999) 625; J. Fleischer, O.V. Tarasov, M. Tentyukov, *Nucl. Phys. Proc. Suppl.* 89 (2000) 112; F. Jegerlehner, M.Y. Kalmykov, O. Veretin, *Nucl. Phys. B* 641 (2002) 285
J. Fleischer, et al., [hep-ph/0202109](#);
D.I. Kazakov, V.N. Velizhanin, *Phys. Rev. D* 65 (2002) 085041;
A. Onishchenko, O. Veretin, *Phys. Lett. B* 551 (2003) 111; M. Awramik, et al., *Phys. Rev. D* 68 (2003) 053004; J.H. Kuhn, et al., *Phys. Rev. D* 68 (2003) 033018
T. Hahn, et al., [hep-ph/0307132](#);
J. Fleischer, et al., *Eur. Phys. J. C* 31 (2003) 37; F. Jegerlehner, M.Y. Kalmykov, *Acta Phys. Polon. B* 34 (2003) 5335
A.I. Onishchenko, V.N. Velizhanin, [hep-ph/0309222](#).
- [3] M. Tentyukov, J. Fleischer, *Nucl. Instr. and Meth. A* 502 (2003) 570.
- [4] J. Fleischer, F. Jegerlehner, O.V. Tarasov, *Nucl. Phys. B* 566 (2000) 423.
- [5] J.A.M. Vermaseren, [math-ph/0010025](#).
- [6] J. Gluza, et al., *Nucl. Instr. and Meth. A*, (2004) these proceedings.
- [7] M. Tentyukov, J. Fleischer, [hep-ph/0311111](#).