Computer Physics Communications 225 (2018) 1–9

Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

CPC 50th anniversary article

Loopedia, a database for loop integrals

C. Bogner [a], S. Borowka [b], T. Hahn [c,*], G. Heinrich [c], S.P. Jones [c], M. Kerner [c], A. von Manteuffel [d], M. Michel [e], E. Panzer [f], V. Papara [c]

[a] Institut für Physik, Humboldt-Universität zu Berlin, D–10099 Berlin, Germany
[b] Theoretical Physics Department, CERN, Geneva, Switzerland
[c] Max-Planck-Institut für Physik, Föhringer Ring 6, D–80805 Munich, Germany
[d] Department of Physics and Astronomy, Michigan State University, East Lansing, MI 48824, USA
[e] CP3, Université Catholique de Louvain, B–1348 Louvain-la-Neuve, Belgium
[f] All Souls College, University of Oxford, OX1 4AL, Oxford, UK

ARTICLE INFO

Article history:
Received 27 September 2017
Received in revised form 8 December 2017
Accepted 21 December 2017
Available online 28 December 2017

Keywords:
Loop integrals
Database
WWW

ABSTRACT

Loopedia is a new database at `loopedia.org` for information on Feynman integrals, intended to provide both bibliographic information as well as results made available by the community. Its bibliometry is complementary to that of Inspire or arXiv in the sense that it admits searching for integrals by graph-theoretical objects, e.g. its topology.

1. Introduction

Researchers in high-energy physics enjoy nearly unsurpassed (in both quality and freedom of access) bibliographic access and resources through well-maintained databases such as Inspire and arXiv. For the practitioner in loop calculations one recurring problem is, however, that the existing databases are indexed solely by 'traditional' metrics: author, title, year of publication, etc. It is not even possible to formulate queries of the kind "Find publications which refer to loop integral X", where X is specified in some graph-theoretical way, say by its topology.

Loopedia attempts to fill this gap, though it is not limited to bibliographic information. The description field of each record can hold any kind of textual information (e.g. URLs to software), and in addition arbitrary files can be uploaded, for example Fortran programs or Maple worksheets.

What we presently provide and describe herein is the database software only, with a few entries for illustration. Populating the database with content is a job both beyond the capabilities of a trained librarian and ultimately in the interest of the researchers who computed the loop integrals, to make their contributions easily findable and get cited.

Integrals are generally associated with their underlying graph. In the case of non-trivial numerators, linear combinations, or set integrals for a sector, the reference is indexed by the corner (scalar) integral of the sector (topology). Tensor integrals (in the one-loop case) are intentionally not covered; justifiable exceptions can be put with the scalar integral, with a suitable explanation.

2. Prolegomena

For the purposes of Loopedia an L-loop Feynman integral is usually an object of the form

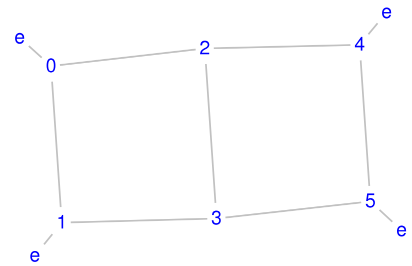
$$\int \frac{\mathrm{d}^D\ell_1 \cdots \mathrm{d}^D\ell_L}{(k_1^2 - m_1^2)^{\nu_1} \cdots (k_n^2 - m_n^2)^{\nu_n}}, \tag{1}$$

where the k_i are linear combinations of the loop momenta $\ell_1, \ldots, \ell_L$ and the graph's external momenta $q_1, \ldots, q_E$, dimensionally regularized in D dimensions. (Exceptions to this form can be accommodated as long as the integral possesses a corresponding Feynman graph.)

Feynman loop integrals are rendered as graphs and as such can be identified by a representation of the corresponding graph. A common choice is the *edge (adjacency) list:* each propagator (edge) is specified by the pair of vertices (nodes) it connects — external nodes need not be distinguished to identify the topology. For example, the following double box has the edge list (e,0) (0,1) (0,2) (e,1) (1,3) (2,3) (2,4) (3,5) (e,4) (4,5) (e,5):

* Corresponding author.
E-mail address: hahn@mpp.mpg.de (T. Hahn).

https://doi.org/10.1016/j.cpc.2017.12.017
0010-4655/



The Nickel index [1–3] furnishes an alternative representation which is constructed as follows: start at a vertex 0 and write down all vertices connected to 0 (e for external legs), then insert a '|'. Repeat for the other vertices but omit edges already included. For the double box above this procedure gives

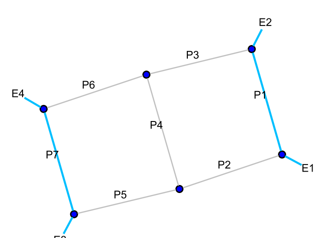
vertex	0	1	2	3	4	5
connected to	*e*12	*e*[0]3	[0]34	[12]5	*e*[2]5	*e*[34]

and hence its Nickel index is `e12|e3|34|5|e5|e|`.

The Nickel index provides a compact identifier and has therefore been chosen as the central object by which integrals are indexed in Loopedia. Neither the edge list nor the Nickel index is unique, however, i.e. depending on the chosen labeling the same graph may have different edge lists/Nickel indices. Loopedia therefore canonicalizes all input.

Besides this 'bare' Nickel index, which represents just the graph's topology, Loopedia introduces an augmented form, the colored Nickel index, *CNickel* for short, which additionally captures the layout of masses and external q^2. Loopedia color-codes them on screen:

Nickel `e12|e3|34|5|e5|e|`
CNickel `e12|e3|34|5|e5|e|:110|10|00|0|11|1|`



For each 'edge identifier' of the Nickel (e or a digit), the coloring has a mass identifier:

0 for zero,
{z, n, s} for {any, non-zero, special} mass scale,
1...9, a...y (not n, s) for a definite non-zero mass scale.

The 's' option is for cases like thresholds or pseudo-thresholds, and the exact meaning of 'special' should be noted in the description of the integral.

The 'definite' identifiers also represent arbitrary scales (though not zero) but unlike the 'zns' choices they express equality of scales within the graph, like named patterns in a computer-algebra system. The graph in the example above has several massive propagators/legs, but all with the same scale '1'.

3. User guide

Loopedia is located at

`loopedia.org`

and upon access, the Start Page™ (Fig. 1) is displayed. One enters a graph in the upper input field and/or chooses constraints for the search with the lower controls and then hits the [Search] button.

The graph is entered via its edge list. The typist has some latitude there: generally the input is understood as long as a pairing of vertices into propagators can reasonably be made out. This means that e.g. FeynArts [4] or QGRAF [5] notation can directly be pasted into the input form. An edge list is internally converted to a Nickel index right away. Alternately, the Nickel index can also be entered directly.

All [C]Nickel indices are first brought into a standard form (for details see Section 4.3). Because the standard form involves amputation of a tadpole's single leg, searching for one leg automatically redirects to searching for zero legs.

3.1. Browsing graphs

The search results are displayed in Loopedia's Graph Browser™ (Fig. 2). The matches are organized so that the topology appears on the left side and the configurations for which information is available are on the right. Every graph icon is clickable.

Clicking on one of the graph icons opens the single-graph display for that graph. For icons in the left column it starts the Configuration Editor™ (Fig. 3), for icons in the right column it opens the Record Display™ (Fig. 6). The Graph Browser is skipped and the Configuration Editor started immediately if the search turns up no matches and an explicit graph was entered, to commence adding a new record.

3.2. Viewing single graphs

The navigation panel at the top has buttons for editing the configuration [Edit • : •], the topology [Edit •], browsing the topology's configurations [Browse • : *], and returning to the start page [Home]. Depending on the situation not all buttons may be active.

In the main part of the page, a picture of the graph is shown underneath its various textual representations (edge list, [C]Nickel index, database path). To the right is the Configuration Chooser™, with the defaults preset to the current values (if any).

From time to time a graph will have an 'ugly' rendering. If it is the first diagram of its kind (i.e. no records added yet) the Configuration Editor will feature a 'Redraw' button on top of the graphic (Fig. 4) to draw it again with a different random seed. When entering new graphs, please allow your aesthetic subconscious to press that button if necessary!

The menu items refer to the labeling of the graph: P1, P2, etc. for internal propagators and E1, E2, etc. for external legs. Fig. 5 shows the drop-down menus expanded out. Observe that the 'remove P*n*' option in the mass menu is somewhat of an anomaly as it removes the corresponding propagator, i.e. it changes not the colored but the bare Nickel index.

Each m_i listed in the menus represents an arbitrary non-zero scale. Unlike the 'any' choice they allow to express equality of scales within the graph, like named patterns in a computer-algebra system. The actual names are insignificant and will in general be renumbered in the colored Nickel index constructed from the HTML form. All different m_i is the most general case; choosing identical m_i or zero refers to a special case. For a loop integral with three propagators, for example, any of the following combinations identifies the case of the first two masses equal: (m_1, m_1, m_2), (m_2, m_2, m_1), (m_2, m_2, m_3), (m_3, m_3, m_2), (m_1, m_1, m_3), (m_3, m_3, m_1).

When editing a topology (bare Nickel index only) the page ends here, i.e. in order to proceed one needs to choose a configuration first. Otherwise the page continues with the available records (Fig. 7) and the New Record Form™ (Fig. 8).

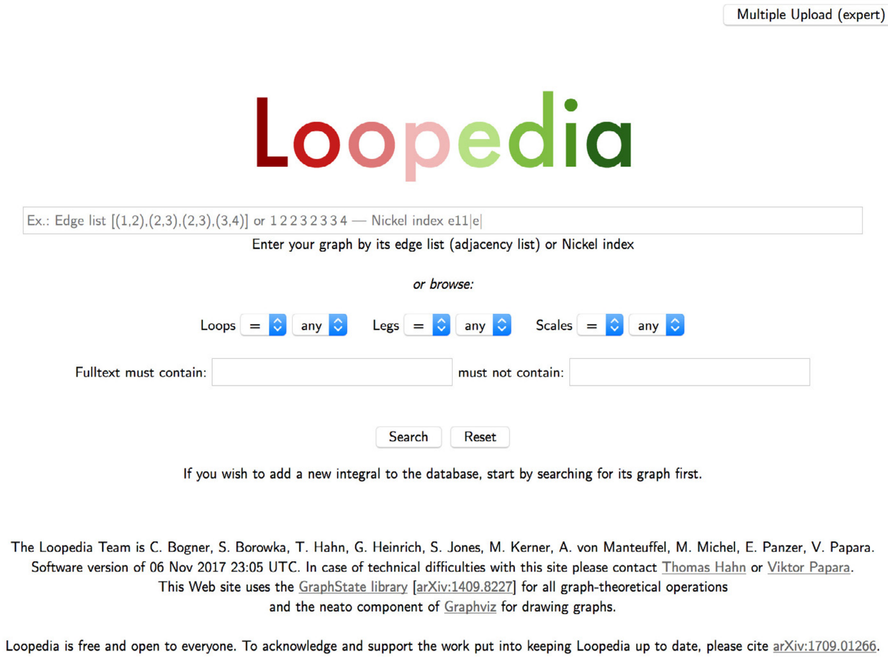


Fig. 1. The Loopedia start page.

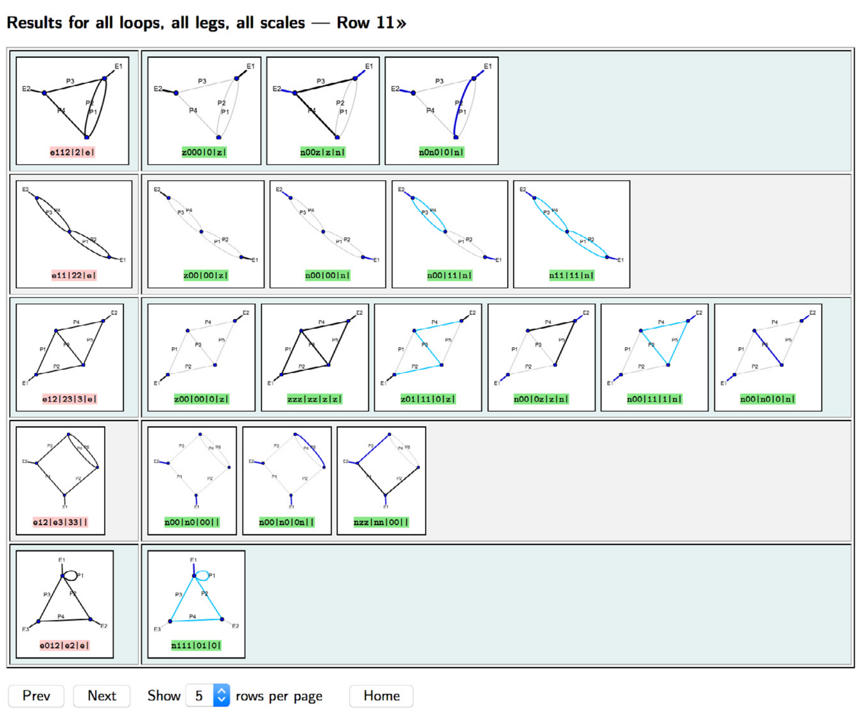


Fig. 2. The Loopedia graph browser.

3.3. Entering new records

To enter a new record one must first navigate to a graph and configuration as outlined above. The New Record Form is found beneath the available records. Its fields are largely optional although at least either the Reference or Description field must be non-empty.

While a few of the input fields ask for specific information on the integral we have found ourselves unable to come up with a (reasonably concise) list of fields that would cover all cases, given the diversity of strategies and concepts followed in the computation of loop integrals. For this reason a special importance falls onto the Description field as general-purpose free-text field. Things we encourage submitters to put in the description field include

- Detailed bibliographic information. State in which equation or section the integral is given. Provide hints about auxiliary information, e.g. definitions relevant to the integral. Point to ancillary files or resources, e.g. included in the arXiv submission. In general: try to make the reader understand

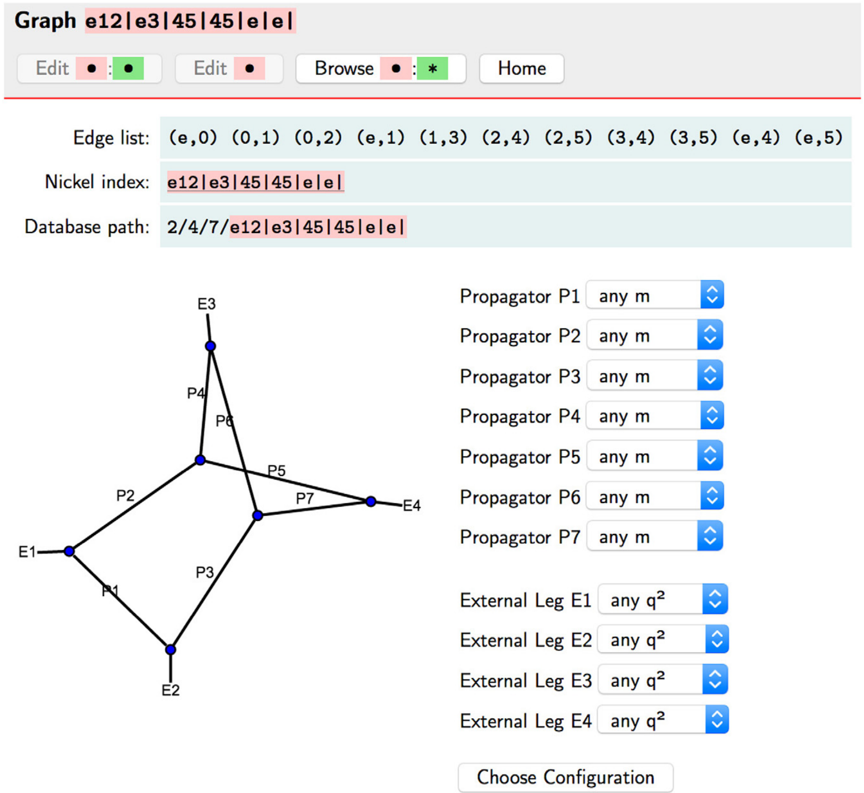


Fig. 3. The Loopedia configuration editor.

what the paper provides and whether it could be relevant to his problem.

- When referencing software packages: The platform(s) they are available for. In which form(at) the results are given, e.g. symbolic expression, subroutine, table of numbers.
- The structure the solution possesses. What kind of functions it is comprised of. What range of validity it has.
- Whether the entry represents linear combinations, sets of integrals, or non-standard propagators. If so, please also fill the dedicated field for the number of master integrals considered in the reference.
- The normalization used in the reference. For example, this normalization will often include factors like μ^{4-D}/Π from the integral measure, where it is relevant whether Π is defined as $i\pi^{D/2}$ or $(2\pi)^D$ (or something else).

The input fields allow some minor formatting:

- special characters are entered in HTML, e.g. "`π`" for π,
- text in `$...$` is displayed in a font mimicking TeX's math mode,
- the `b` in `A_b`, `A_{b}`, `A^b`, `A^{b}` is sub- (`_`) or superscripted (`^`),
- URLs and arXiv references are automatically made clickable.

If you have machine-readable results, e.g. Mathematica notebooks or FORM files, upload them even if they duplicate the results in the paper cited.

Upon successful submission the submitted record is displayed again for review, with controls for editing and deletion (Fig. 9). An e-mail is sent to the submitter's address with an individualized URL that allows to access the review page again for future editing.

The submitted records are not publicly visible until confirmed by a moderator. This is a safety policy we implemented for now to forestall arguments, such as submitters deleting each

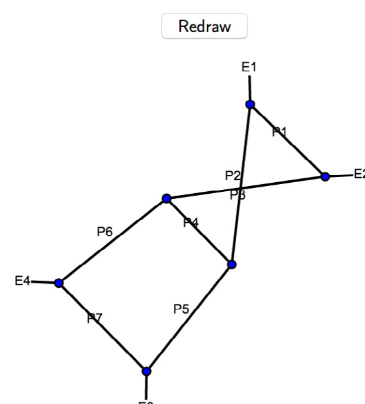


Fig. 4. An 'ugly' graph and the redraw button.

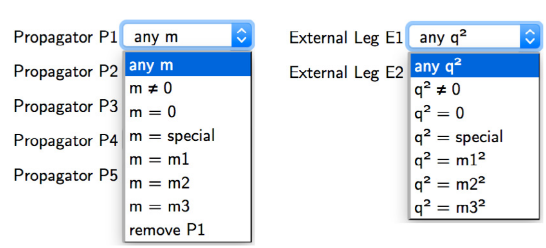


Fig. 5. Drop-down menus for graph configuration.

other's records, and may change in the future. The moderators are notified together with the submitter and will usually make new records public in a timely manner. Moderators may make

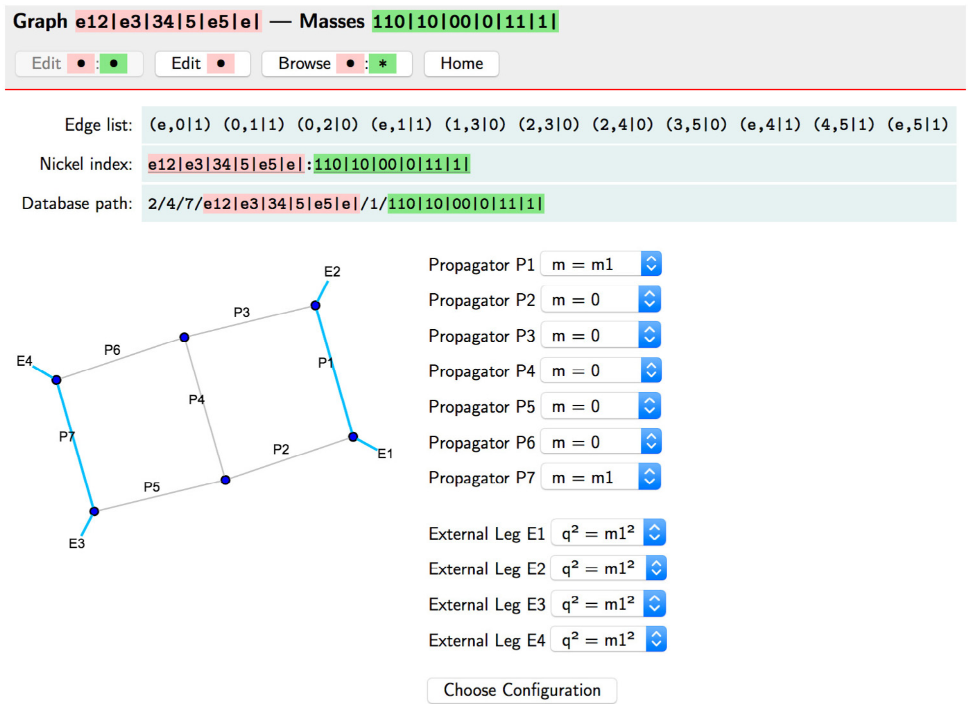


Fig. 6. The Loopedia record display.

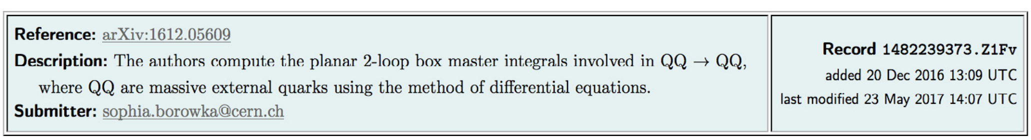


Fig. 7. Example of a single record listing.

changes at their discretion, though the present editing policy is that substantial changes require contacting the submitter, if an e-mail is given. If you wish to become a moderator please contact `loopedia@mpp.mpg.de`.

The records belonging to a particular CNickel index are kept in one of three 'visibility' bins, `private`, `public`, and `deleted`, where (non-privileged) database lookups only access records in `public`. A new submission starts life in `private`, moderator confirmation moves it to `public`, deleting it to `deleted`, and a submitter edit back to `private`. The moderators' superpowers consist in being able to (a) view all bins, (b) move records freely between the bins, and (c) remove them completely, too. All of these actions are logged.

3.4. Multiple upload

A situation commonly encountered in practice is that several integrals are computed in one paper and when adding those one would of course like to avoid re-typing the reference, authors, description, etc. Loopedia aids this in two ways.

Firstly by 'remembering' information: a record once entered is 'carried over' to the next graph, i.e. the submission form comes with the fields already filled out. Unless one wants to add details specific to the new graph, one can just click Submit.

Secondly, Loopedia has a special mode for multiple uploads, found in the top right corner of the start page: Multiple Upload (expert). It requires that the CNickels of the graphs are known (though not necessarily normalized), or alternately an edge list with the mass identifiers as third members is given, as in: `((e,0,p) (0,1,1) (0,1,1) (e,1,p))`. This is particularly attractive if a program exists to generate those.

The form for multiple submission is pretty much the same as the standard New Record Form except that the graph and any non-standard propagator powers are entered in the box at the top (Fig. 10). Per line one graph is entered, followed by any non-standard powers in the form `P5=2`. The ε-order and the number of masters can be overridden for that line by respectively adding e.g. `eps=-2,-1` or `nmasters=15`.

The submit button does not immediately add the records to the database in this case but performs a 'dry run' first (Fig. 11).

4. Internals

4.1. Design decision

Initially we considered a fairly 'standard' design with an SQL backengine interfaced through a PHP-based frontend but moved away from this design for several reasons, most importantly long-term maintainability. In a science world where software projects often languish after the student who wrote them leaves the field we wanted our database and frontend to fulfill three requirements:

- *Maintenance and portability to a different platform should be simple and require only minimal system administration.* SQL for one needs quite some administrator intervention to set it up because it brings along its entire ecosystem. And at least binary portability of the tables is not a given as different systems/distributions come with different SQL implementations.

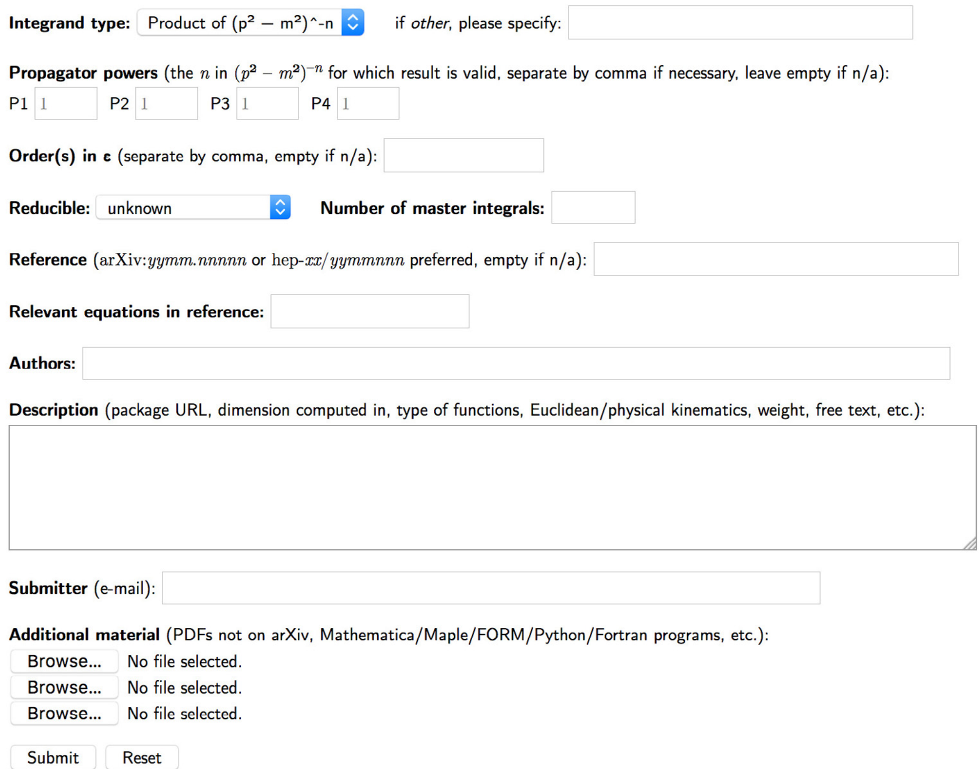


Fig. 8. The new record form.

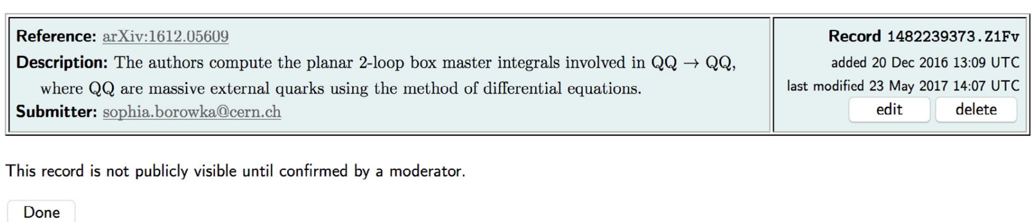


Fig. 9. The Loopedia record review.

- *The database entries should be accessible by standard Unix tools.*
  This makes inspection and fixing of problems much simpler, and furthermore it is less of a burden to the backup since, after each modification, only the files that actually changed need to be backed up, not the entire SQL table.
- *The realization should not be based on a software framework which requires frequent updates, in particular security ones.*
  For example, one framework we looked at was Drupal, with two major versions being maintained and a projected lifetime of 8–10 years for a release. Drupal versions are in general not upward compatible; assuming we joined somewhere in the middle of a release, this means having to rewrite after about 5 years, not counting updates in relevant third-party packages, which typically have no life-cycle management.

Finally we settled on a bash script in a CGI environment which uses the Unix file system as database, indexed by the `mlocate` utility, and fulfills all three criteria above.

4.2. Database structure

The Loopedia database contains (in a very real filesystem sense) one directory for each bare Nickel index, with subdirectories for the configuration index (the second part of the CNickel) of the same graph. Underneath comes another layer of subdirectories for the visibility (`public`, `private`, `deleted`), and below that, one directory for each record (Fig. 12). Doubling the filesystem as a database is not all that unusual; Apple, for example, chose a similar design for its iTunes music database.

The actual implementation inserts a few more directory layers for indexing and performance reasons so that the full database path becomes

db/L/ℓ/p/Nickel/s/Config/Visibility/Record

where L, ℓ, p, s are the number of loops, legs, propagators, and scales, respectively. The database root 'db' is included in the path to be able to make anchored searches.

Records have identifiers similar to Unix's maildir format, e.g. `1482239373.Z1Fv`, which is the Unix time of submission (seconds after 1/1/1970) followed by a `mkdtemp` suffix to avoid race conditions.

Indexing of the database is performed with the `mlocate` Unix utility. The latter indexes the file name only but since that includes the extra directory layers just mentioned we can also search for the number of loops, for example. With a little tweak in the source code we were able to make `mlocate` work with relative paths, which

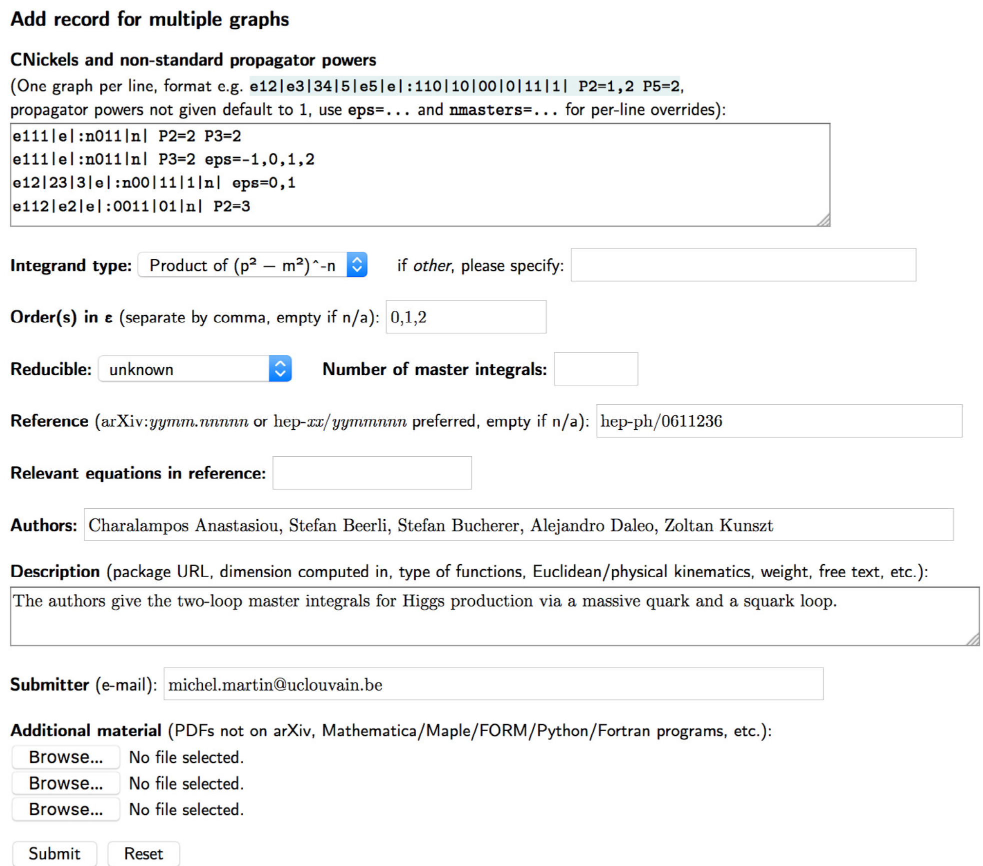


Fig. 10. The multiple upload form.

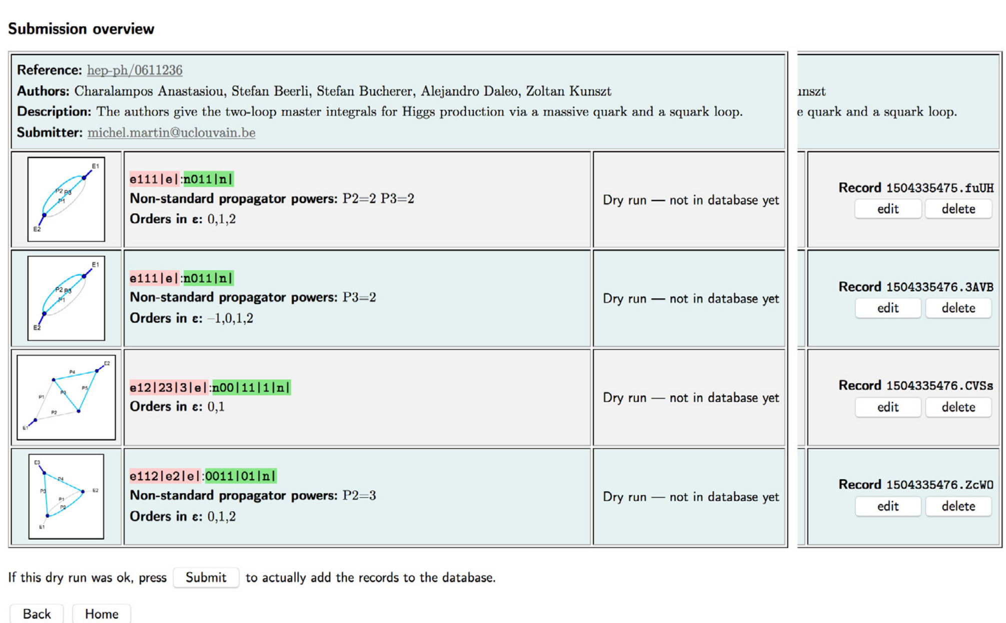


Fig. 11. Dry run and final submit for multiple upload.

means its output is exactly of the form above and the index file is independent of the database's location in the file tree.

Unix filesystems are actually pretty powerful databases but are neither made for nor really expected to scale in extreme cases. A common bottleneck is excessively many files in one directory — a case which in Loopedia is efficiently mitigated by the extra directory layers introduced above. Loopedia is currently running on Linux's Ext4 journaling filesystem.

4.3. Graph-theoretical operations

Almost all graph-theoretical operations in Loopedia are taken care of by the Python library GraphState [3]. GraphState allows to supplement a graph with configurations, i.e. add extra integers to either the nodes or the edges. The colored Nickel index used by Loopedia implements a single edge coloring which encodes masses and external q^2.

GraphState's default output format for a configuration is decimal, with underscores delimiting the integers, for example `0_0_0_0|0_0_0|38|`. We wanted the coloring to have the same length as the bare Nickel index, however, since that makes visual pairing of the bare Nickel index (left column in Graph Browser) and its colorings (right column) particularly straightforward, and also identifiers like '38' (which might stand for 'any mass scale') are not very self-explanatory. Therefore we generalized GraphState's formatting routines slightly, to allow the CNickel notation described earlier.

GraphState automatically brings each graph into a normal form but does not touch the coloring as it has no notion of what it represents. We therefore added a routine to canonicalize the entire CNickel so that no spurious duplicates would swamp the database. Operations include: a tadpole's single leg is cut off since it carries no information for the integral, single instances of definite scales $(1, 2, \ldots)$ are converted to 'any non-zero scale' (`n`), remaining definite scales are renumbered, for two-point functions momentum conservation is enforced (i.e. equal identifiers for incoming and outgoing legs).

Lastly, we added a Python function to process user input and bring it into the form of an edge list acceptable to GraphState. Mostly we strip function heads (i.e. replace $f(i)$ by i) and if the remaining list has no substructure, partition it in pairs of two. This means that the input for the same diagram can be as diverse as

- `1 3 2 4 3 4 3 4`
- `[(1,3) (2,4) (3,4) (3,4)]`
- `Topology[2][`
  `Propagator[Incoming][Vertex[1][1],`
  `Vertex[3][3]],`
  `Propagator[Outgoing][Vertex[1][2],`
  `Vertex[3][4]],`
  `Propagator[Loop[1]][Vertex[3][3],`
  `Vertex[3][4]],`
  `Propagator[Loop[1]][Vertex[3][3],`
  `Vertex[3][4]] ]`

A QGRAF output style suitable for Loopedia would be

```
<prologue>
<diagram>
<propagator_loop>(<vertex_index>,
<dual-vertex_index>) <end>
<epilogue>
<exit>
```

4.4. Drawing graphs

The most appealing representation of a graph for humans is still its picture, and indeed both the Graph Browser and the Single-Graph Display take advantage of this. Automatically drawing an arbitrary graph (with reasonable output) is a difficult business, however, which we gladly leave to the 'neato' component of the Graphviz package [6]. The shapes may not always be the ones traditionally associated with Feynman diagrams but the important point here is that the propagator routing and labeling can be understood at a glance with as few ambiguities as possible.

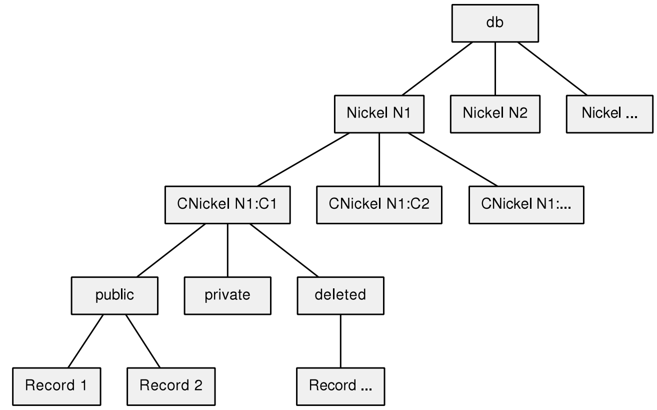


Fig. 12. Structure of the Loopedia database.

Behind the scenes a Python function translates the graph into Graphviz's DOT language, from which neato produces an SVG image. SVG is an XML-based vector format rendered by all modern browsers and scales well in the icon-size plaquettes of the Graph Browser. All images are laid out with the same absolute font sizes and length scales but may appear with different weights relative to each other in the Graph Browser due to being scaled to fit.

The graph corresponding to a colored Nickel index is really colored. In particular the massless case, drawn in gray with reduced line weight, can instantly be recognized even in the small image sizes of the Graph Browser.

4.5. HTTP and CGI handling

The 'heart chamber' of Loopedia is a bash script named `index.cgi`. It interacts with the Web server through the Common Gateway Interface (CGI). Also CGI is known to suffer from common exploits through unguarded handling of the CGI query strings but we have two tough little C programs for processing.

For the default type of HTML form (`application/x-www-form-urlencoded`) we use our `unescape` utility, which parses the CGI input into variable assignments suitable for bash's `eval` command. Needless to say, the left-hand sides are suitably sanitized and the right-hand sides properly quoted to prohibit execution of remote code.

HTML forms emanating from `index.cgi` submit only with the `POST` method, mainly so as not to mess up the user's URL field, but in the e-mail to submitter and moderators a `GET`-type URL is sent for direct referral to a particular record, and of course `unescape` can deal with any combination of `POST` (stdin) and `GET` (environment) input.

For the record submission we must use the `multipart/form-data` type since it may include upload of (binary) files. This kind of input is handled by `formdecode`; it splits the input stream along the delimiters and stores the results in files, in a temporary directory. Obviously we have to sanitize these filenames, too, but want to keep the original form as much as security allows, for there is usually some meaning at least in the extension (`.pdf`, `.f`, `.m`) if not in the entire name.

Except in the e-mails, `index.cgi` at no point references an absolute URL — in fact, the HTML forms are completely self-referential as there is only one CGI script around. Not only does this make testing easy (the script runs without change on 'localhost' on a developer's laptop, for example) but we can also manage privileged access simply by symlinking the Loopedia root directory to a different directory for which we require authentication through the Apache configuration. Elevated rights are granted if the server presents a valid user.

Submitters gain restricted privileges to modify their own submission through a URL including a token. The random 32-character alphanumeric token is generated at submission and e-mailed to the submitter. It both identifies the record it belongs to uniquely and grants permission to edit or delete that record.

Loopedia's outward appearance is governed by the `loopedia.css` style sheet and degrades gracefully if the CSS takes too long to load or is blocked. CSS and the generated HTML code validate cleanly (no errors, no warnings) on {Firefox, Chrome, Safari, Opera} $\otimes$ {Linux, MacOS}.

5. Summary

In this paper we presented a new database for bibliographic and other information on loop integrals. Loopedia collects the available information and makes it searchable by graph. We hope that Loopedia will in time be able to answer the query "Find all papers pertaining to graph X".

The database is hosted at the Max Planck Institute for Physics in Munich with broadband internet access and a daily backup at `loopedia.org`.

It is now up to the loop-integral calculators to make this effort a success — and *en passant* attract more citations for their work.

Acknowledgments

We thank Claude Duhr and Valentin Hirschi for important discussions. C. Bogner thanks Deutsche Forschungsgemeinschaft for support under the project BO 4500/1–1. S. Borowka gratefully acknowledges financial support by the ERC Advanced Grant MC@NNLO (340983) and ERC Starting Grant "MathAm" (39568) during different stages of this project. M. Michel is supported by the "Fonds Spécial de Recherche" of the Université catholique de Louvain and grateful to the TH department at CERN for hospitality during various stage of this project. This research was supported in part by the Research Executive Agency (REA) of the European Union under the Grant Agreement PITN-GA2012316704 (HiggsTools). We gratefully acknowledge support by the Munich Institute for Astro- and Particle Physics (MIAPP) of the DFG cluster of excellence "Origin and Structure of the Universe".

References

[1] J.F. Nagle, J. Math. Phys. 7 (1966) 1588.
[2] B.G. Nickel, D.I. Meiron, G.A. Baker, Compilation of 2-Pt and 4-Pt Graphs for Continuous Spin Model, University of Guelph Report, 1977.
[3] D. Batkovich, Y. Kirienko, M. Kompaniets, S. Novikov, arXiv:1409.8227. The GraphState library is at https://pypi.python.org/pypi/GraphState.
[4] J. Küblbeck, M. Böhm, A. Denner, Comput. Phys. Commun. 60 (1990) 165;
T. Hahn, Comput. Phys. Commun. 140 (2001) 418. hep-ph/0012260.
[5] P. Nogueira, J. Comput. Phys. 105 (1993) 279.
[6] E.R. Gansner, S.C. North, Softw. - Pract. Exp. 30 (2000) 1203. graphviz.org.