Kira: Integral Reduction

(common work with: Fabian Lange, Philipp Maierhöfer)

High Precision for Hard Processes (HP2 2022)

Johann Usovitsch

22. September 2022

Outline

1. Introduction of Kira
   - Introduction of Laporta algorithm
   - Introduction to Feynman integral reduction language
   - General seeding procedure in Kira
2. Main bottleneck in Kira 4-loop reductions
   - Solution 1: improved seeding
   - Solution 2: sectors are linearly independent
3. Automatic permutation of propagators
4. Premature extension of Kira to 6-loop reductions
5. Construction of the block triangular form
6. Summary and outlook

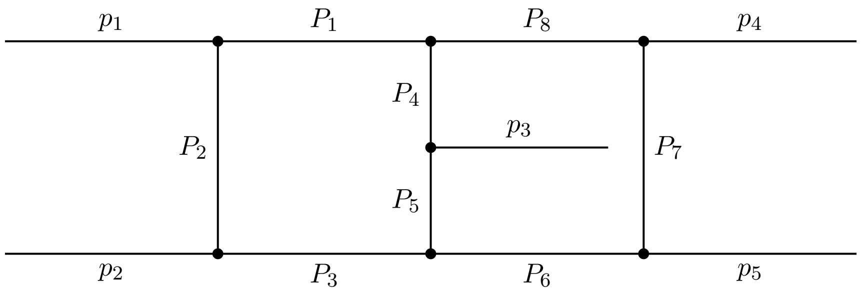
General features of Kira

- MPI support
- Finite field support
- Reduction of general linear system of equations
- Automatic generation of IBPs and symmetry finder for multiple integral toplogies

General purpose of Kira

- Reduction of $2 \to 2$ doubleboxes (first application in single top production at t-channel)
- Reduction of $1 \to 2$ three-loop form factors (first application $H \to gg$ 3-loop form factor)
- Application of user defined systems
  - Gradient flow formalism [R. V. Harlander, F. Lange, 2022]
  - Phase-space integrals with Heaviside functions [D. Baranowski, M. Delto, K. Melnikov, C.-Y. Wang, 2021]
  - Solving system of differential equations (collaboration with Martijn Hidding on DiffExp [Hidding, 2020], used by AMFlow [Xiao Liu,Yan-Qing Ma, 2022])
  - Double-pentagon topology in five-light-parton scattering (solve block triangular form: [Xin Guan, Xiao Liu, Yan-Qing Ma, 2019])

Double-pentagon topology in five-light-parton scattering



Runtime	Memory	Probes	CPU time per probe	CPU time for probes
12 d	540 GiB	38278000	0.37 s	25 %

- Including d, the reduction of the double-pentagon topology is a six variable problem
- We use a system of equations which is in **block-triangular form** taken from [Xin Guan, Xiao Liu, Yan-Qing Ma, 2019], which is of the size of 72 MB, best value I could find comparing to other methods. And no simplifications where yet applied.
- We benchmark the reduction of all integrals including five scalar products

Double-pentagon topology in five-light-parton scattering

- Block-triangular form degree of the polynomials is 7
- From [JU, 2020] one denominator coefficient in the IBP table

$$(-8+d)*(-6+d)^3*(-5+d)^3*(-4+d)^3*(-3+d)^2*(-2+d)*(-1+d)*(-11+2*d)*(-9+2*d)*(-7+2*d)*s15^2*(s15-s23)*s23^4*(1+s15-s34)^5*(s15-s23-s34)^4*(-1+s34)*s34^6*(-1+s45)^4*s45^3*(-1-s23+s45)^3*(s15-s23+s45)^4*(-1+s34+s45)^5*(s34+s45)^2*(-1+s34+s45+s34*s45)*(s15-s23+s23*s34-s15*s45+s34*s45)*(-s15+s23-s23*s34-s45+s15*s45-2*s23*s45+s45^2)*(1+s23-s34-2*s23*s34-2*s45-s23*s45+s34*s45+s45^2)*(-(s15*s34)+s23*s34-s23*s34^2+s15*s45-s23*s45-2*s34*s45-s15*s34*s45-s23*s34*s45+s34^2*s45-s15*s45^2+s34*s45^2)*(s15^2-2*s15*s23+s23^2+2*s15*s23*s34-2*s23^2*s34+s23^2*s34^2-2*s15^2*s45+2*s15*s23*s45+2*s15*s34*s45+2*s23*s34*s45+2*s15*s23*s34*s45-2*s23*s34^2*s45+s15^2*s45^2-2*s15*s34*s45^2+s34^2*s45^2)$$

- One term after the expansion: $8d^{17}[l^{59}]s15^2s23^5s34^{21}s45^{31}$

Integration-by-parts (IBP) identities

$$I(a_1,\ldots,a_5)=\int\frac{d^Dl_1d^Dl_2}{[l_1{}^2-m_1^2]^{a_1}[(p_1+l_1)^2]^{a_2}[l_2{}^2]^{a_3}[(p_1+l_2)^2]^{a_4}[(l_2-l_1)^2]^{a_5}}$$

$$\int d^D\boldsymbol{l_1}\ldots d^D\boldsymbol{l_L}\frac{\partial}{\partial(\boldsymbol{l_i})_\mu}\left((q_j)_\mu\frac{1}{[P_1]^{a_1}\ldots[P_N]^{a_N}}\right)\quad \text{[Chetyrkin, Tkachov, 1981]} \quad =0$$

$$c_1(\{a_f\},\vec{s},D)I(a_1,\ldots,a_N\boldsymbol{-1})+\cdots+c_m(\{a_f\},\vec{s},D)I(a_1\boldsymbol{+1},\ldots,a_N)\quad =0$$

$q_j=p_1,\ldots,p_E,l_1,\ldots,l_L \quad \vec{s}=(\{s_i\},\{m_i^2\})$
m number of terms generated by one IBP identity

Reduction: express all integrals with the same set of propagators but with different exponents a_f as a linear combination of some basis integrals (master integrals)

- Gives relations between the scalar integrals with different exponents a_f
- a_f = integers: Sample a system of equations, **Laporta algorithm** [Laporta, 2000]
- Public tools: Kira [Klappert, Lange, Maierhöfer, Usovitsch, Uwer, 1705.05610, 2008.06494], Reduze2 [von Manteuffel, Studerus, 1201.4330], FIRE [Smirnov, Chuharev, 1901.07808], FiniteFlow [Peraro, 1905.08019]+LiteRed[Lee, 1310.1145]

Integral seeds

- **Topology** is the name of your integral family with a linearly independent set of propagators
- **Seeds** are integrals with integer power coefficients, e.g: topo7[1,1,1,1,1,1,2,-2,-1]
- To generate system of equations we apply IBP identities to the seeds
- $r = \sum_{j=1}^{N} a_j\,\theta(a_j - \frac{1}{2})$ sum of positive indices (example 8)
- $s = -\sum_{j=1}^{N} a_j\,\theta(-a_j - \frac{1}{2})$ sum of negative indices (example 3)
- Dots $d = \sum_{j=1}^{N} a_j\,\theta(a_j - \frac{3}{2})$ (example 1)
- Sector number $S = \sum_{j=1}^{N} 2^{j-1}\,\theta(a_j - \frac{1}{2})$
- "Number of lines" is the number of propagators with positive exponent power

Seeding in Kira

- Suppose we are interested in the reduction of the topology topo7 with $s = 4$ and $r = 7$
- In Kira we generate seeds for 7-line integrals with sector 127 $r = 7$, $d = 0$, $s = 4$
- We also generate seeds for 6-line integrals with sectors [63, 126,...] $r = 7$, $d = 1$, $s = 4$
- We also generate seeds for 5-line integrals with sectors [62, 124, ...] r=7, $d = 2$, $s = 4$
- ...

Bottleneck

- Important: forward elimination is dominating the run time in a finite field reduction
- Generating system of equations in the style of Kira is a major bottleneck with growing number of loops
- Because we are keeping reduction parameters s and d constant for all sectors
- But: for each bottleneck in Kira we always find a hack

Improved seeding

- Seed integrals in Kira for 7-line integrals with sector 127 $r = 7$, $d = 0$, $s = 4$
- For 6-line integrals with sectors [63, 126,...] $r = 7$, $d = 0$, $s = 3$
- For 5-line integrals with sectors [62, 124,...] $r = 7$, $d = 0$, $s = 2$, ...
- One reduction sample of the system of equations over one finite field will give $\tilde{N}$ of $\{\tilde{M}_i\}$ master integrals
- We generate additional IBP of the seeds which come from $\{\tilde{M}_i\}\backslash\{M_i\}$, where $\{M_i\}$ are the minimal set of master integrals
- Perform the reduction again and see if more seeds are required
- Right now it is tedious to do these steps in Kira, but we plan to make this hack a feature based on ideas from Mao Zeng

Sectors are linearly independent in the forward elimination

- If we do not use the option preferred masters, then the linear independance of sectors is observed for all topologies, as soon as we have linearly independent set of equations
- This should give a big improvement if we perform the Gaussian elimination for each sector individually and only perform the backward substitution to all sectors
- I believe the improvement is significant, and the new bottleneck for the run time of a finite field reduction is the reconstruction of final coefficients or the backward substitution
- This is already implement within the option of `run_triangular: sectorwise`

New option

- Add `permutation_option: 1` in integralfamilies.yaml
  - Physical propagators first
  - Propagators with least number of terms first
  - Zero mass propagators first
- Specify your own ordering with `permutation: [3,2,1,4,5,6,7,8,9]`
- We have 4-loop examples where the perfomance improvement is measured to be about a factor of 300

Change of internal limits

- Kira soon supports integrals with up to 63 propagtors
- We change the limits of the algorithm for the seed integral compactification
- Soon each seed is stored in a unique identification number of 128 bit size instead of 64, you will find a warning which will remind you, that some things go slowlier.
- Number of equations possible to reduce with Kira increases to 2^{64} from 2^{32}

Construction of the block triangular form, see arXiv:1912.09294v3

- First step is the Ansatz: $I_1 c_1 + \cdots + I_N c_N = 0$, where I_i are the Feynman integrals and the c_i are polynomials.
- Second step is the Ansatz for the coefficients
$$c_j(d, \vec{s}) = \sum_{i=0}^{d_{\max}} d^i \sum_{\vec{l} \in \Omega_{k_j}}^{k_{\max}} \hat{c}_j^{i, l_1, \ldots, l_M} s_1^{l_1} \cdots s_M^{l_M}$$
- $\Omega_{k_j} = \{\vec{l} \in \mathbb{N}^M | \sum_{j=0}^{M} l_j = k_j\}$
- We have a linear relation between integrals of different massdimension, thus k_i differ with respect to the integrals of our choice
- The $\hat{c}_j^{i, l_1, \ldots, l_M}$ are unknown rational numbers and are fixed by adjusting the $k_{\max}$ and $d_{\max}$

Block-triangular form

- To determine the unknowns $\hat{c}_j^{i,l_1,\ldots,l_M}$ we have to reduce the IBP-system to N master integrals generated the Laporta way as many times as the number of the unknowns $\hat{c}_j^{i,l_1,\ldots,l_M}$ are in the Ansatz.
- Each new sample generates N new non trivial equations.
- Some unknowns turn out to be $\hat{c}_j^{i,l_1,\ldots,l_M}$ undetermined and we can choose them arbitrary.
- **The result** is a system of equations in block triangular form containing as many equations as integrals, which we would like to reduce.
- The coefficients are polynomials of very low degree
- The rational numbers $\hat{c}_j^{i,l_1,\ldots,l_M}$ will be huge
- This system of equations is ideal for the finite field methods applied in Kira
- We have a working general implementation of this algorithm
- We are looking for an appropriate hack to address the bottleneck associated with integrals with dots

Upcoming Features in next Kira Version

Kira's, development release

Get Kira on gitlab: https://gitlab.com/kira-pyred/kira.git

- On https://hepforge.kira.org we provide a static linked Kira executable
- We have a Wiki and a best practice summary on gitlab
- We plan to go for the block triangular form: `run_triangular: block`, which finds a small and fast to evaluate system of equations for general topologies [Xin Guan, Xiao Liu, Yan-Qing Ma, 2020]!
- We have automated the permutation of propagators to accelerate the reduction time `permutation_option: 1`
- We improved the speed for the export of the results into the FORM output
- More dedicated improved seeding

Summary and Outlook

- Kira is an all-rounder for multi-scale as well as for multi-loop computations
- Kira utilize the finite field methods and helps to tailor it to your needs
- Computing the block triangular form will allow us to tackle new interesting state of the art problems!
- Explained few bottlenecks in 4-loop computations

Finite field reconstruction: Kira + FireFly

- Reconstruction of multivariate rational functions from **samples over finite integer fields** [Schabinger, von Manteuffel, 2014][Peraro, 2016]
- Public implementations available: FireFly [Klappert, Lange, 2019][Klappert, Klein, Lange, 2020], FIRE 6 [Smirnov, Chukharev, 2019] and FiniteFlow [Peraro, 2019]
- **FireFly has been combined with Kira's native finite field linear solver**
- Furthermore Kira supports MPI: to utilize the new parallelization opportunities now available with finite field methods
- **Side note:** the collaboration [Dominik Bendle, Janko Boehm, Murray Heymann, Rourou Ma, Mirko Rahn, Lukas Ristau, Marcel Wittmann, Zihao Wu, Yang Zhang, 2021] implements semi-numeric row reduced echelon form. They play with Laporta ordering in intermediate steps to improve the reduction time for the forward elimination!

Run time examples

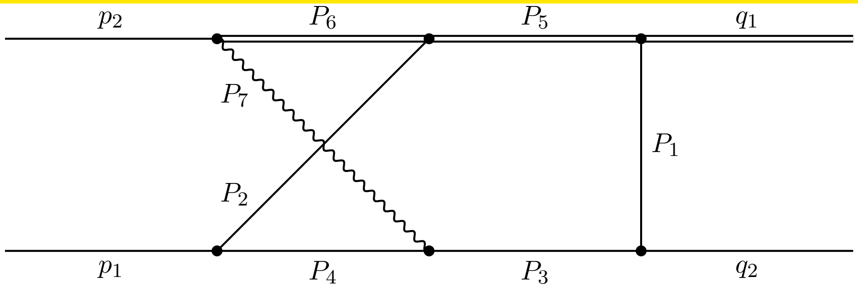
$$P_1 = k_1^2, \quad P_2 = k_2^2, \quad P_3 = k_3^2, \quad P_4 = (p_1 - k_1)^2, \quad P_5 = (p_1 - k_2)^2, \quad P_6 = (p_1 - k_3)^2, \quad P_7 = (p_2 - k_1)^2,$$
$$P_8 = (p_2 - k_2)^2, \quad P_9 = (p_2 - k_3)^2, \quad P_{10} = (k_1 - k_2)^2, \quad P_{11} = (k_1 - k_3)^2, \quad P_{12} = (k_2 - k_3)^2,$$

$$p_1^2 = z z_b, \quad p_2^2 = 1, \quad p_1 p_2 = (1 - z)(1 - z_b)$$

We chose $r = 17$ and $s = 0$ for the benchmark

Mode	Runtime	Memory	Probes	CPU time per probe	CPU time for probes
`run_initiate`	5 h 20 min	128 GiB	-	-	-
`run_triangular` + `run_back_substitution`	$>$ 14 d	~ 540 GB	-	-	-
`run_firefly: true`	6 d 3 h	670 GiB	108500	370 s	100 %
`run_triangular: sectorwise`	36 min	4 GiB	-	-	-
`run_firefly: back`	4 h 54 min	35 GiB	108500	12.2 s	100 %

Reducing the memory footprint with iterative reduction



$r = 7$ and $s = 4$

Mode	Iterative	Runtime	Memory
`Kira` $\oplus$ `FireFly`	-	18 h	40 GiB
	sectorwise	33 h 15 min	9 GiB

- `iterative_reduction: sectorwise` — one sector at a time
- `iterative_reduction: masterwise` — one master integral at a time
- Works well with the options `run_back_substitution` and `run_firefly`
- Independent study confirms the efficiency of this method

  [Chawdhry, Lim, Mitov, 2018]
- Sacrifice the CPU time for 4 times less main memory consumption

Runtime reduction with coefficient arrays

`--bunch_size=`	Runtime	Memory	CPU time per probe	CPU time for probes
1	18 h	40 GiB	1.73 s	95 %
2	14 h	41 GiB	1.30 s	94 %
4	11 h	46 GiB	1.00 s	93 %
8	10 h 15 min	51 GiB	0.91 s	92 %
16	9 h 45 min	63 GiB	0.85 s	92 %
32	9 h 30 min	82 GiB	0.84 s	92 %
64	9 h 30 min	116 GiB	0.83 s	92 %
`Kira` $\oplus$ `Fermat`	82 h	147 GiB	-	-

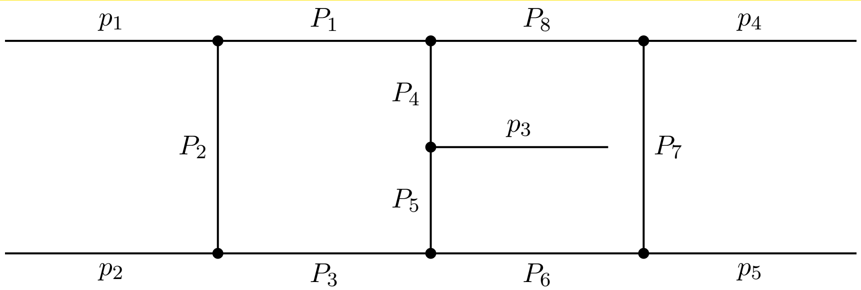
- The runtime of the probes is dominated by the forward elimination
- 48 cores each with hyper-threading disabled
- Coefficient arrays bring sizeable effects in exchange for main memory

Runtime reduction with MPI

# nodes	Runtime	Speed-up	CPU efficiency
1	18 h	1.0	95 %
2	10 h 15 min	1.8	87 %
3	7 h 15 min	2.5	82 %
4	5 h 45 min	3.1	76 %
5	5 h 30 min	3.3	65 %
Kira ⊕ Fermat	82 h	-	-

- Option `run_firefly: true` and `Intel`® `MPI` is used
- The first prime number suffers in the performance because FireFly cannot process arbitrary probes
- New probes are scheduled based on intermediate results
- **Remark:** the user should use less nodes for the first prime number

Double-pentagon topology in five-light-parton scattering I



Runtime	Memory	Probes	CPU time per probe	CPU time for probes
12 d	540 GiB	38278000	0.37 s	25 %

- Including d, the reduction of the double-pentagon topology is a six variable problem
- We use a system of equations which is in **block-triangular form** taken from [Xin Guan, Xiao Liu, Yan-Qing Ma, 2019], which is of the size of 72 MB, best value I could find comparing to other methods. And no simplifications where yet applied.
- We benchmark the reduction of all integrals including five scalar products

Double-pentagon topology in five-light-parton scattering II

- FireFly's factor scan improves the denominators
- `-bunch_size = 128` option is used to improve the speed
- 40 cores with hyperthreading enabled
- The most complicated master integral coefficient has a maximum degree in the numerator of 87 and in the denominator of 50
- The database of the reduction occupies 25 GiB of disk space
- The number of required probes 10^7 is computed fast due to the block triangular structure of the system of equations

  [Xin Guan, Xiao Liu, Yan-Qing Ma, 2020]
- Main memory reduction can be achieved with the options `iterative_reduction` or by reducing the `-bunch_size` option
- We use Horner form to accelerate the parsing for the coefficients

Double-pentagon topology in five-light-parton scattering III

- The new option `insert_prefactors` would give a factor of 2 improvement in an overall performance if we use the denominators from [J.U, arXiv:2002.08173]. The method to compute these denominators is explained shortly in the summary of [J.U, arXiv:2002.08173], which relies on algebraic reconstruction methods pioneered in [arXiv:1805.01873, arXiv:1712.09737, arXiv:1511.01071]. A second approach to compute the denominator functions should be possible with finite field methods [Heller, von Manteuffel, arXiv:2101.0828].
- The **block triangular form** is much better suited for the reduction than a naïv IBP system of equations as generated by Kira
- Reduction tables are available upon request