Forcer: a FORM program for 4-loop massless propagators

Takahiro Ueda
Nikhef, The Netherlands

Collaboration with:
Ben Ruijl, Jos Vermaseren and Andreas Vogt

28 April 2016
LL2016, Leipzig

Introduction

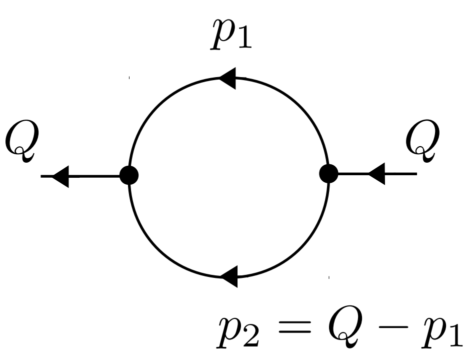
Introduction

- Massless propagator-type Feynman integrals: many physics applications ⇨ Andreas's talk

- At the 1-loop:

$$F(\underbrace{n_1, n_2}) = \int \frac{d^D p_1}{(2\pi)^D} \frac{1}{(p_1^2)^{n_1} (p_2^2)^{n_2}}$$

indices: powers of the denominators



  - Dimensional regularization: $D = 4 - 2\epsilon$
  - The external momenta Q is off-shell: $Q^2 \neq 0$

Feynman integral calculus

- The "standard" way of Feynman integral calculus

  (1) Reduction to master integrals (MIs) via integration-by-parts identities (IBPs) [Chetyrkin, Tkachov '81]

  (2) Evaluation of MIs

- Generic IBP reduction algorithms for any processes
  - Laporta's algorithm, Baikov's method, Lee's "LiteRed", ...
    [Laporta '01] [Baikov '96; '05] [Lee '12; '13]

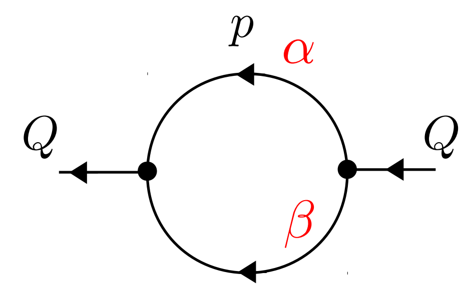
    AIR, FIRE, Reduze,...

- "Mincer" algorithm [Chetyrkin, Tkachov '81]
  - Specialized reduction for massless propagator-type integrals (up to 3-loops), very efficient

“Mincer” approach

General 1-loop formula

- General formula for arbitrary indices α and β



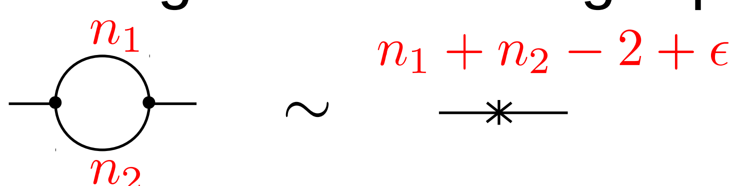
$$\int \frac{d^D p}{(2\pi)^D} \frac{p^{\mu_1} \dots p^{\mu_n}}{(p^2)^\alpha [(Q-p)^2]^\beta}$$ allowing numerators ($n \geq 0$)

$$= \frac{1}{(4\pi)^2} \frac{1}{(Q^2)^{\alpha+\beta-2+\epsilon}} \sum_{\sigma=0}^{\lfloor n/2 \rfloor} G(\alpha,\beta,n,\sigma)(Q^2)^\sigma \left[\frac{1}{\sigma!} \left(\frac{\Box_p}{4} \right)^\sigma p^{\mu_1} \dots p^{\mu_n} \right]_{p=Q}$$

$$G(\alpha,\beta,n,\sigma) = (4\pi)^\epsilon \frac{\Gamma(\alpha+\beta-\sigma-2+\epsilon)}{\Gamma(\alpha)\Gamma(\beta)} B(2-\epsilon-\alpha+n-\sigma, 2-\epsilon-\beta+\sigma)$$

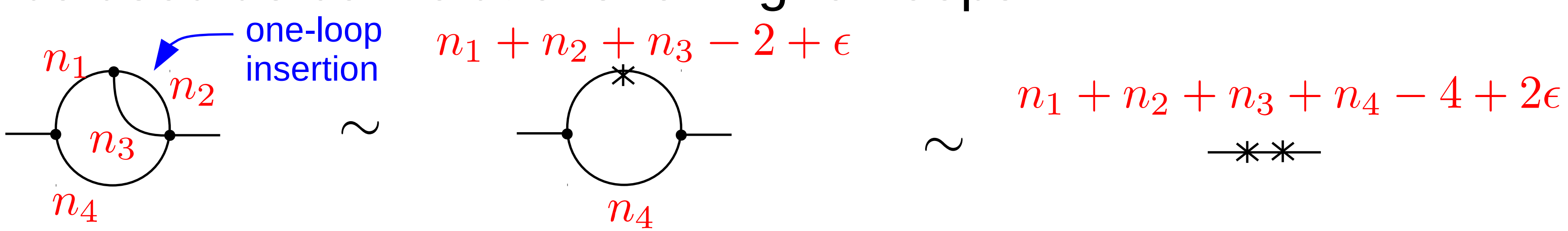
[Chetyrkin, Kataev, Tkachov '80; Chetyrkin, Tkachov '81]

- The result gets a non-integer power $1/(Q^2)^\epsilon$



$\ast$: non-integer part ϵ

- Can be used as convolutions for higher loops

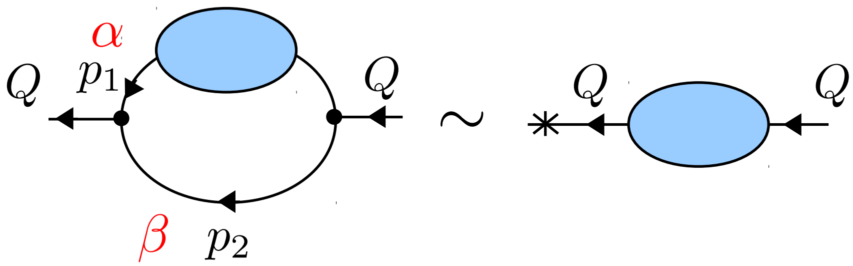


"Carpet" 1-loop integral

- We can perform another type of 1-loop integral (outer loop of p_1) for arbitrary indices α and β possibly with numerators

$p_i \cdot p_j$, $p_i \cdot Q$

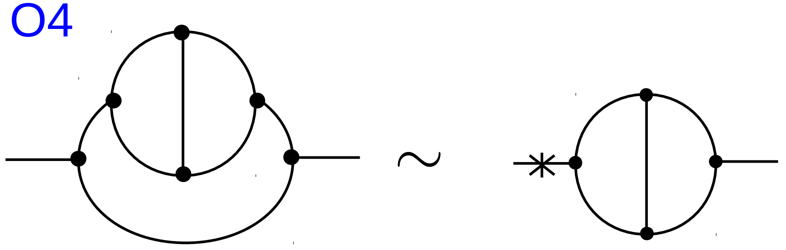
(loop momenta in the blob)



- Used for the following topology at the 3-loop level

[Chetyrkin, Tkachov '81]

O4



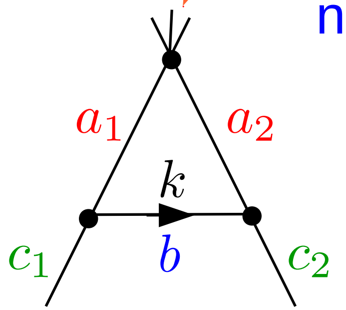
Triangle rule

[Chetyrkin, Tkachov '81]

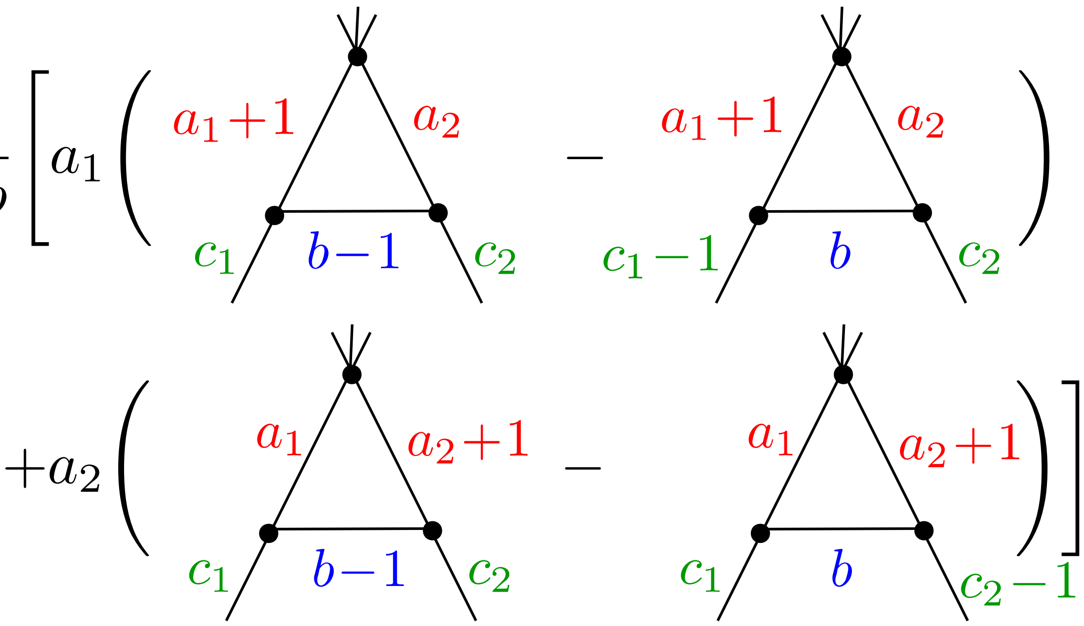
- IBP ($\frac{\partial}{\partial k} \cdot k$) in one-loop triangle-shaped (sub-)diagrams

any # of lines

numerator $k^{\mu_1} \dots k^{\mu_N}$ OK



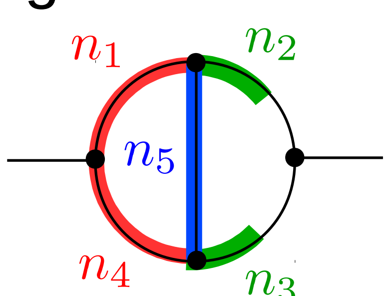
$$= \frac{1}{D + N - a_1 - a_2 - 2b}$$



Decreases b or c_1 or c_2 by 1
at the cost of increasing a_1 or a_2
in the right-hand side

- From positive integer indices, recursive use of the triangle rule makes $b = 0$ or $c_1 = 0$ or $c_2 = 0$ (removal of a line)

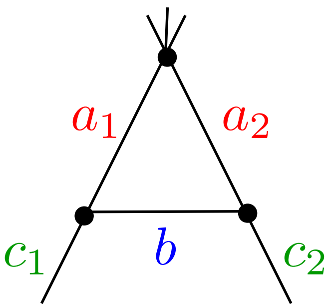
⇒ sums of integrals in simpler topologies



Diamond rule

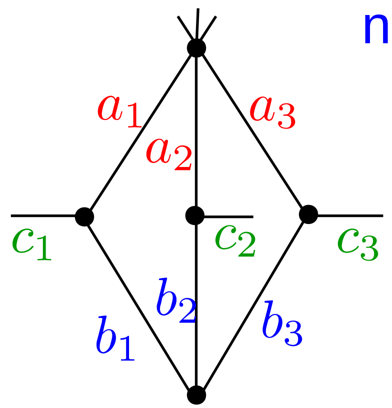
[Ruijl, TU, Vermaseren '15]

- Extension of the triangle rule to multi-loop diamond-shaped (sub-)diagrams



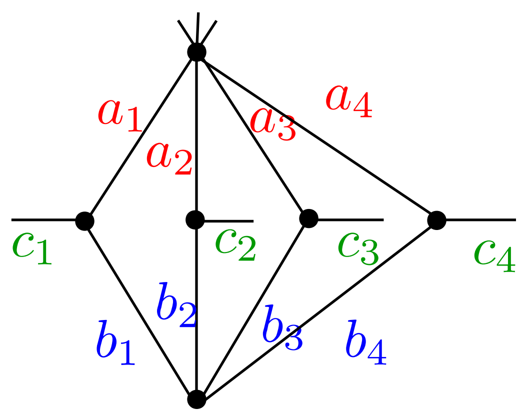
any # of lines

numerator OK



Increases a_1, a_2, a_3

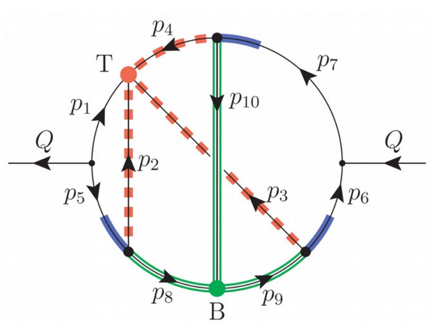
Decreases b_1, b_2, b_3
c_1, c_2, c_3



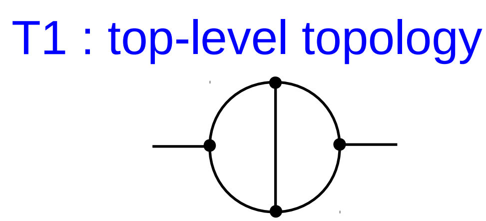
Increases a_1, a_2, a_3, a_4

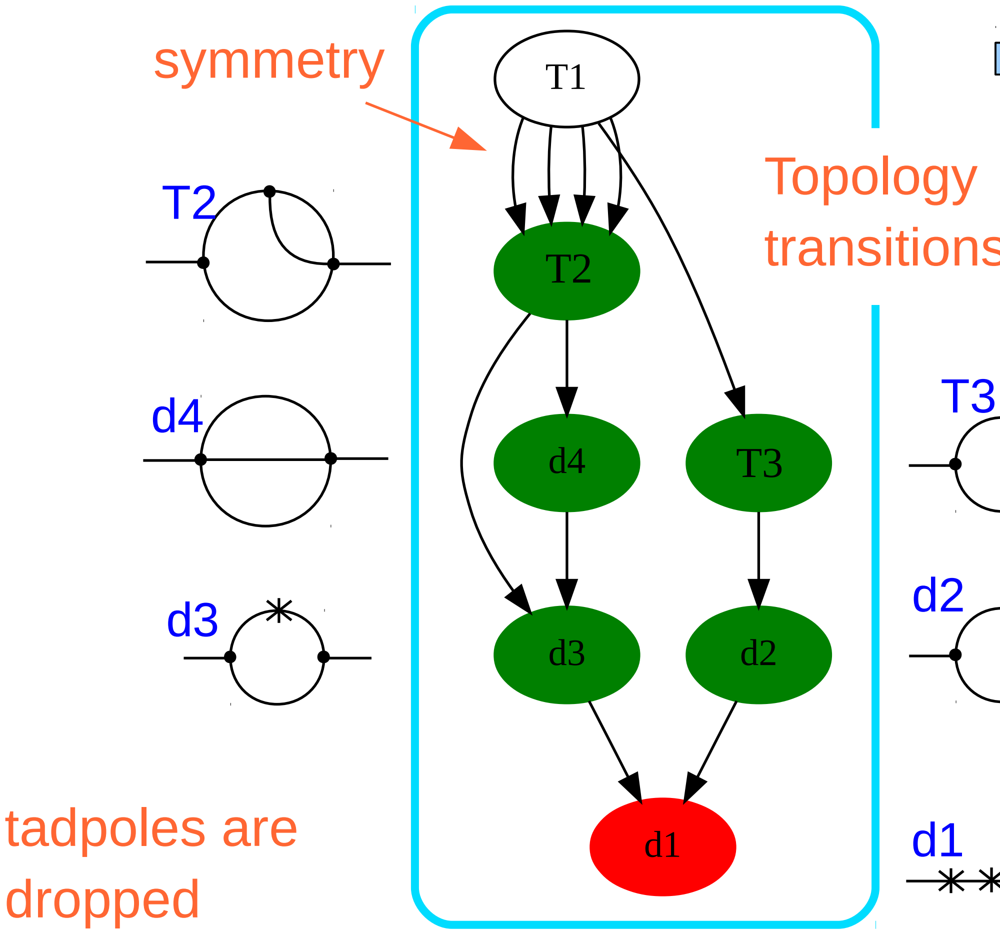
Decreases b_1, b_2, b_3, b_4
c_1, c_2, c_3, c_4

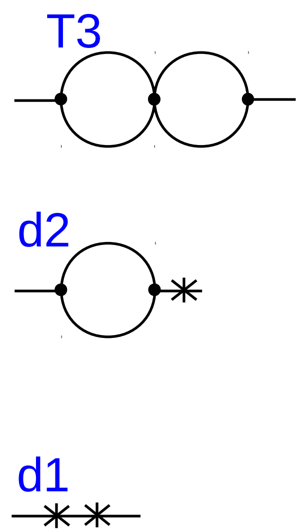
Appear from 4-loops



2-loop topologies







- All integrals can be evaluated by using the triangle rule and performing one-loop integrals

  ⇨ Expressed in terms of Γ -functions

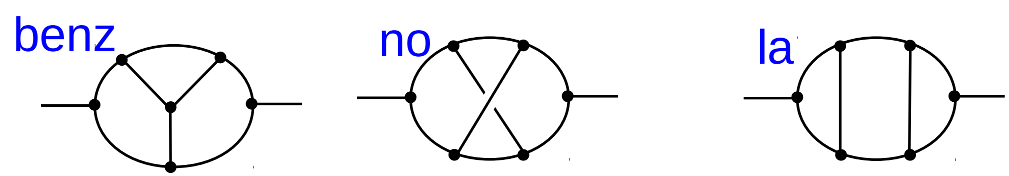
Topologies in which
- One loop-insertion
- Triangle rule

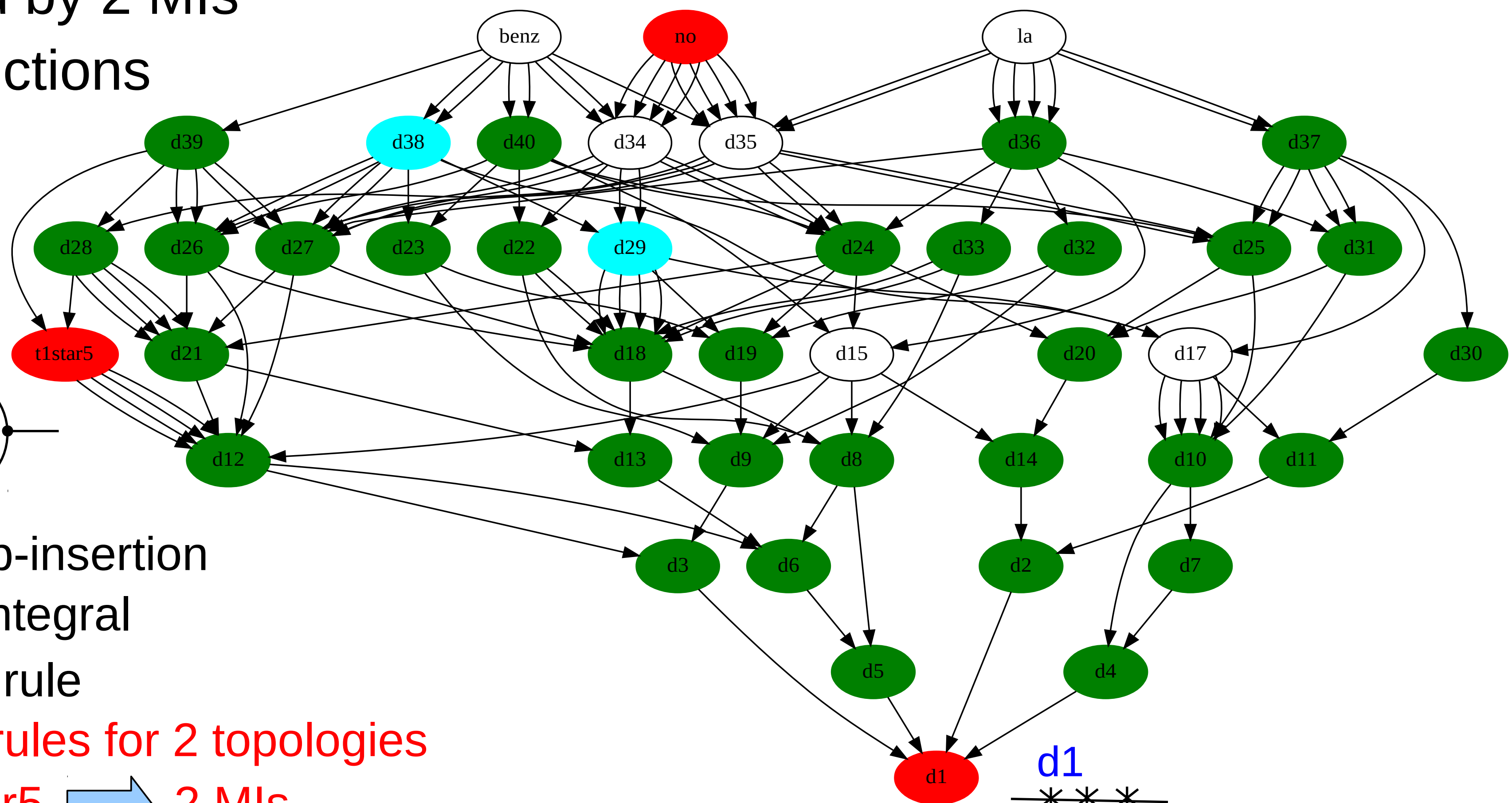
can be applied

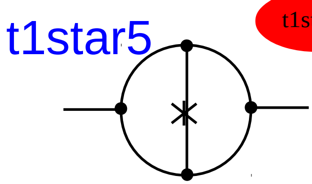
3-loop topologies

- All integrals are expressed by 2 MIs and Γ-functions

3 top-level topologies







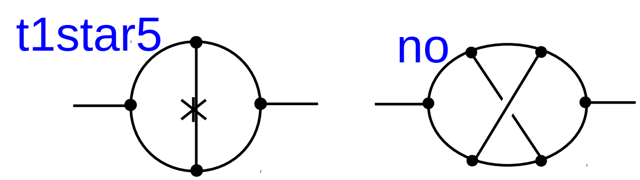
- One loop-insertion
- Carpet integral
- Triangle rule
- Special rules for 2 topologies

no, t1star5 ⇒ 2 MIs

Mincer approach

Algorithm: [Chetyrkin, Tkachov ’81]
Schoonschip implementation: [Gorishny, Larin, Surguladze, Tkachov ’89]
Form implementation: [Larin, Tkachov, Vermaseren ’91]

- Many topologies for massless propagator-type integrals can be reduced to simpler ones by
  - performing one-loop integrals ⇐ makes a big difference from general algorithms
  - use of triangle rules to remove one of lines
- Special cases (where we need to solve IBPs) are not too many
  - Only 2 topologies up to 3-loops



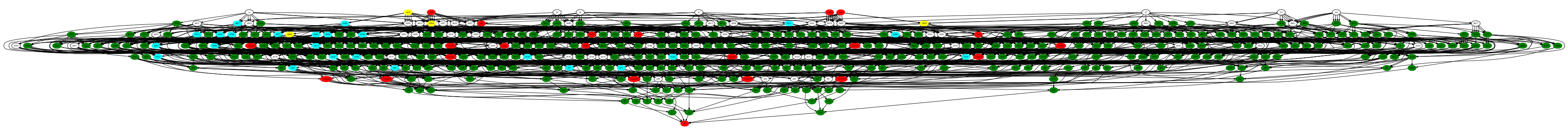
- In principle, this approach can be extended to the 4-loop level
  - Laurent series of the MIs in ϵ at the 4-loop are known

Baikov, Chetyrkin ’10 (via glue-and-cut symmetry)
Lee, Smirnov^2 ’11 (up to weight 12, via DRA, Mellin-Barnes, PSLQ)

Extension to the 4-loop level: “Forcer”

4-loop topologies

11 top-level topologies

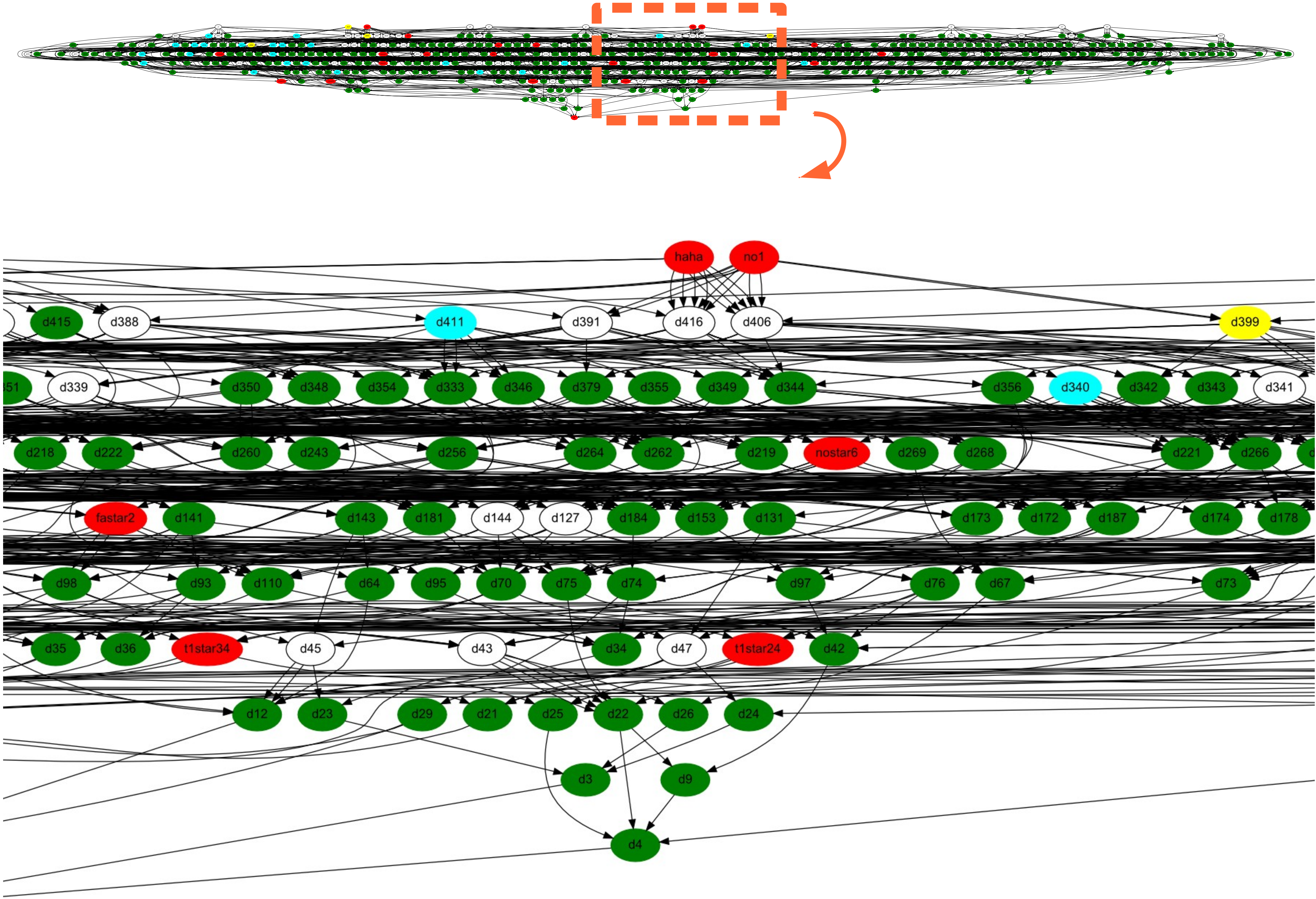


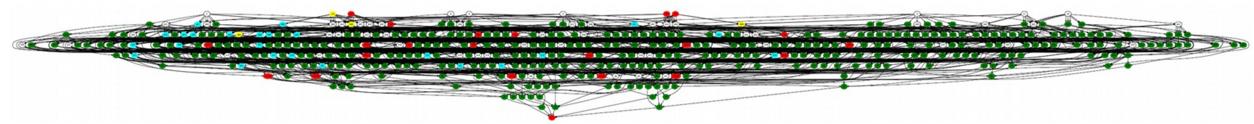
One loop-insertion	(335)	
Carpet integral	(24)	
Triangle rule	(53)	
Diamond rule	(4)	
Special rules	(21)	(pure 4-loop: 9)

Total: 437

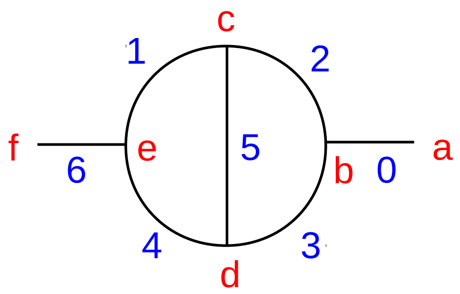
- Problem: enormous number of cases!!
- Coding such a reduction by hand is impractical

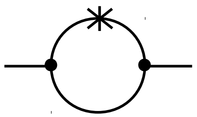
  ⇨ Automatization


haha
no1
d415
d388
d411
d391
d416
d406
d399
d339
d350
d348
d354
d333
d346
d379
d355
d349
d344
d356
d340
d342
d343
d341
d218
d222
d260
d243
d256
d264
d262
d219
nostar6
d269
d268
d221
d266
fastar2
d141
d143
d181
d144
d127
d184
d153
d131
d173
d172
d187
d174
d178
d98
d93
d110
d64
d95
d70
d75
d74
d97
d76
d67
d73
d35
d36
t1star34
d45
d43
d34
d47
t1star24
d42
d12
d23
d29
d21
d25
d22
d26
d24
d3
d9
d4

How to handle 

- Each topology as an "undirected graph" in graph theory

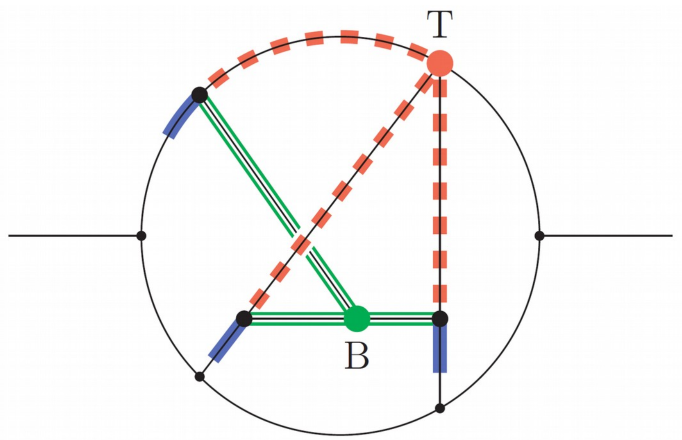


Vertices and edges are labeled
A graph is represented by connections of them
"Stars" as color of edges 

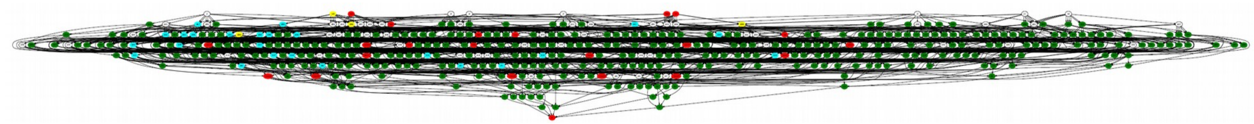
  - Implementation: Python3 with a graph library "igraph"

http://igraph.org

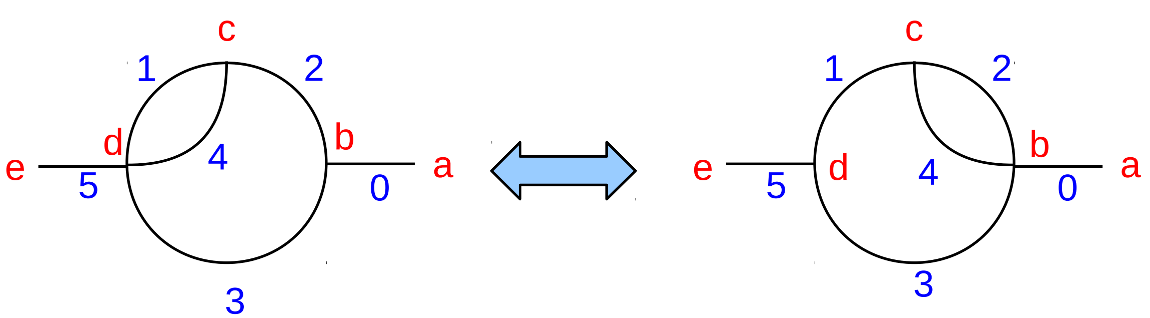
  - Easy to detect one-loop insertion, carpet, triangles, diamonds and tadpoles



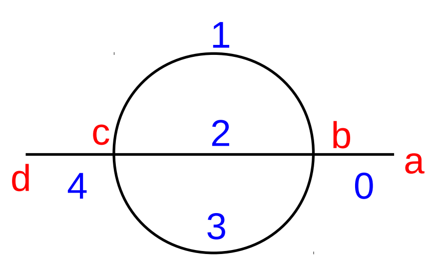
could be difficult by human eyes (diamond example)

How to handle 

- Graph isomorphism
  - Detects equivalent graphs and finds mappings among them

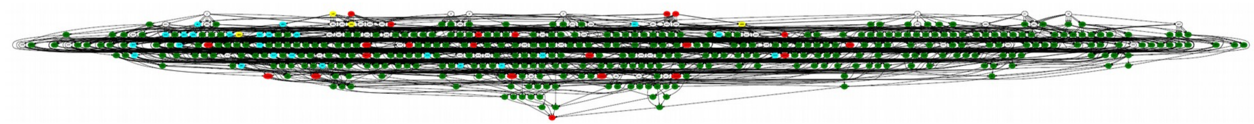


- Graph automorphism
  - Finds symmetry / mappings in each graph

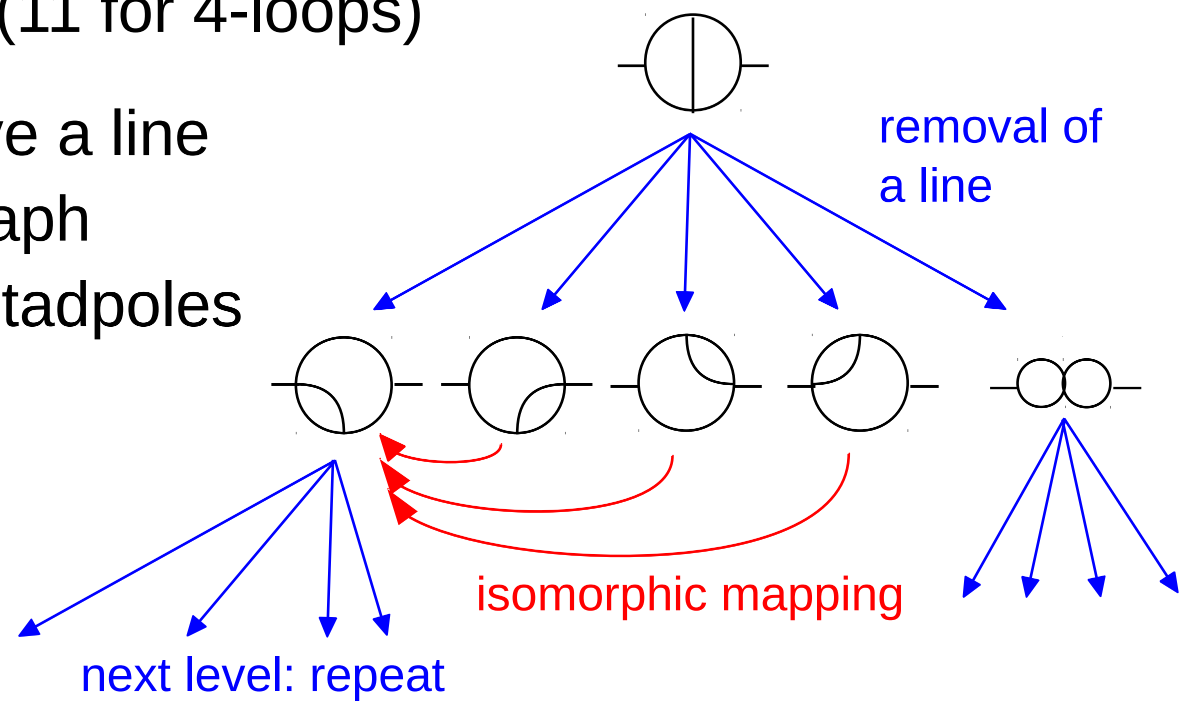


$1 \leftrightarrow 2 \leftrightarrow 3$

How to handle

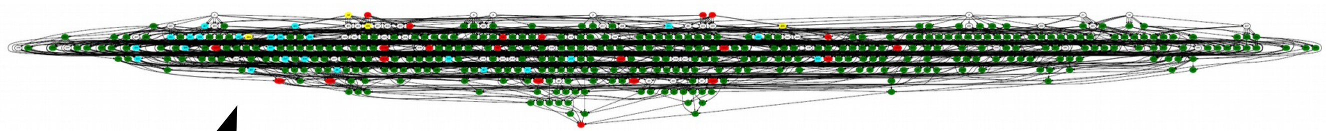


- Input : top-level topologies (11 for 4-loops)
- From each topology, remove a line in all possible ways with graph isomorphism and dropping tadpoles



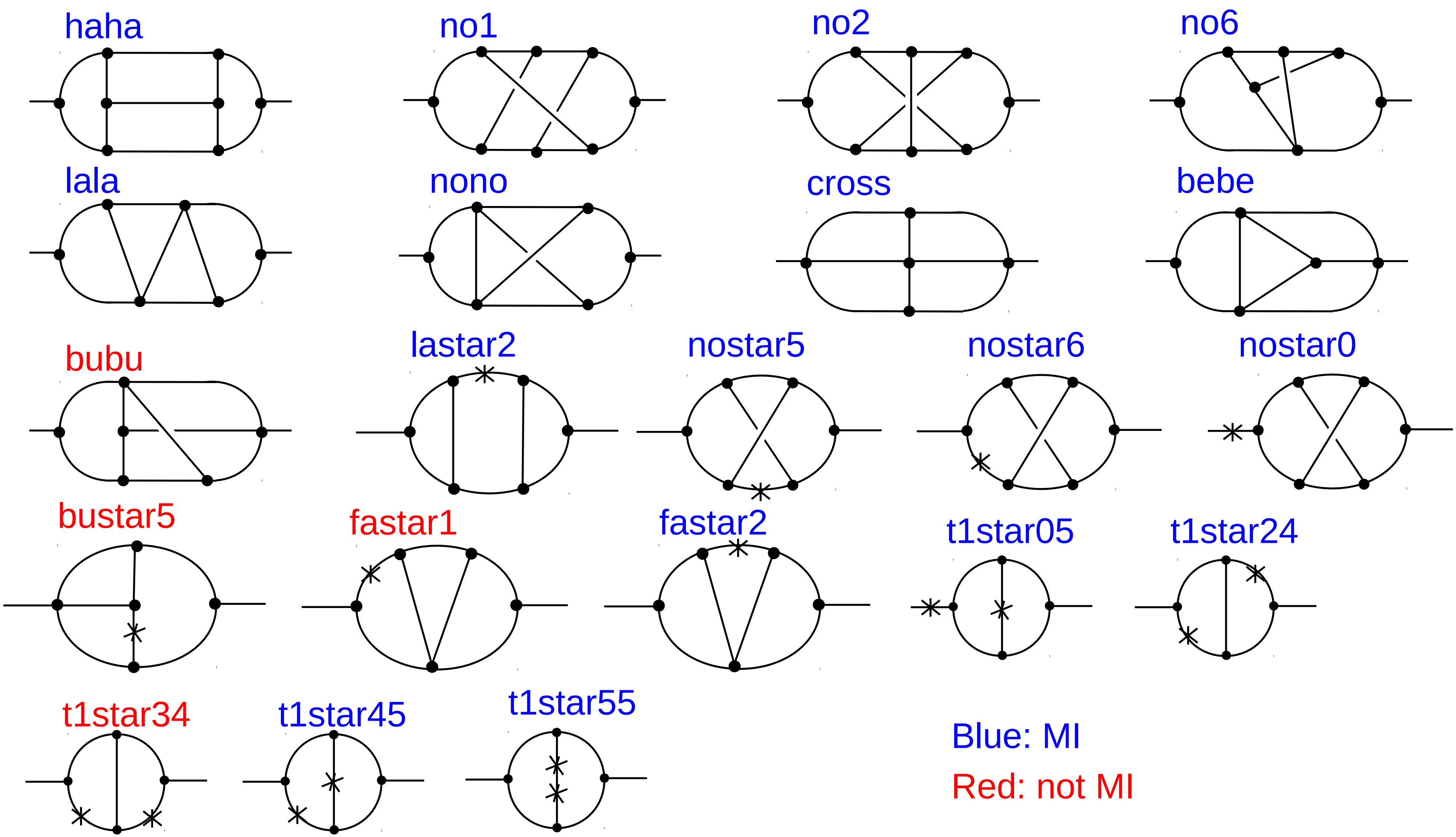
- For each topology, the next action is decided (one-loop, carpet, triangle, diamond, otherwise special rule needed)
  - Irreducible numerators (dot products) are chosen such as they do not interfere with the next action

Code generation

- In the end, we get 
- Code generation from
  - Adequate subroutines are called at each topology
  - Symmetries from graph automorphism
  - Rewriting propagators and irreducible numerators at all transitions from a topology to another
  - Python program with 2813 lines generates FORM code with 39405 lines (+ auxiliary routines) for 4-loops (as of 28.4.2016)
- Works even for 5-loops
  (64 top-topologies, 6570 topology in total, but 284 special rules need to be implemented)

Manual reduction rules

- 21 topologies require special rules, manually constructed



Blue: MI

Red: not MI

Finding manual reduction rules

- Shift an index by 1 (or -1 or more) in IBPs in all possible ways

$$n_1 \to n_1 \pm 1,\ n_2 \to n_2 \pm 1,\ n_3 \to n_3 \pm 1,\ \ldots$$

  - Combined with the original IBPs ⇨ new set of equations

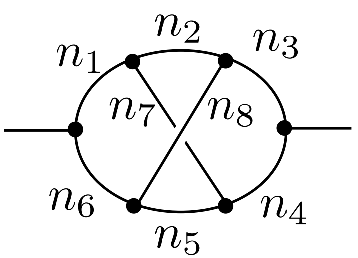
- Eliminate "complicated" integrals that increase indices for propagators (and/or decrease those for irreducible numerators) from the system of equations as possible

- In general, this is not complete, but helps a lot. Human eyes for finding "good" rules for reducing integrals

  less number of terms, decrease of complexity, short coefficients, no spurious poles, etc.

- Expressions may be very complicated. A rule can easily be 100-1000 lines or more. Use of computer algebra systems. (Don't do that by hand!)

Example: 3-loop NO



$n_9 : p_2 \cdot Q$

- 3 loop momenta and 1 external momentum
  8 propagators and 1 irreducible numerator

  ⇨ 3 × 4 = 12 IBPs

- Shift an index by -1 in the IBPs in all possible ways

positive: $n_1, \ldots, n_8$
negative: n_9

$$n_1 \to n_1 - 1,\ n_2 \to n_2 - 1,\ \ldots, n_9 \to n_9 - 1$$

⇨ 12 × 9 = 108 ⇨ 12 + 108 = 120 combined equations

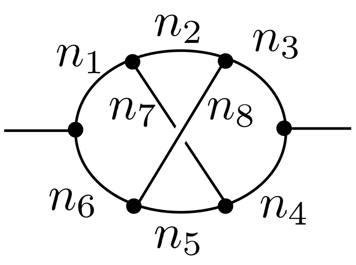
- Eliminate “complicated” integrals from the system of the equations
  A solution is (after overall $n_9 \to n_9 + 1$)

```
id Z(n1?{>=1},n2?{>=1},n3?{>=1},n4?{>=1},n5?{>=1},n6?{>=1},n7?{>=1},n8?{>=1},n9?{<=-1}) =
+Z(n1,n2-1,n3+1,n4,n5,n6,n7,n8,n9+1)*rat(1,2)*rat(n3,-7+n9+n8+n7+n6+n5+n4+n3+n2+n1+4*ep)
+Z(n1,n2,n3+1,n4,n5,n6,n7,n8-1,n9+1)*rat(-1,2)*rat(n3,-7+n9+n8+n7+n6+n5+n4+n3+n2+n1+4*ep)
+Z(n1+1,n2-1,n3,n4,n5,n6,n7,n8,n9+1)*rat(1,2)*rat(n1,-7+n9+n8+n7+n6+n5+n4+n3+n2+n1+4*ep)
+Z(n1+1,n2,n3,n4,n5,n6,n7-1,n8,n9+1)*rat(-1,2)*rat(n1,-7+n9+n8+n7+n6+n5+n4+n3+n2+n1+4*ep)
+Z(n1,n2,n3,n4,n5,n6,n7,n8,n9+1)*rat(1,2)*rat(-3+n9+n3+2*n2+n1+2*ep,-7+n9+n8+n7+n6+n5+n4+n3+n2+n1+4*ep)
+Z(n1,n2-1,n3,n4,n5,n6,n7,n8,n9+2)*rat(1,2)*rat(1+n9,-7+n9+n8+n7+n6+n5+n4+n3+n2+n1+4*ep)
;
```

Repeated use until $n_9 = 0$ (removal of the irreducible numerator)

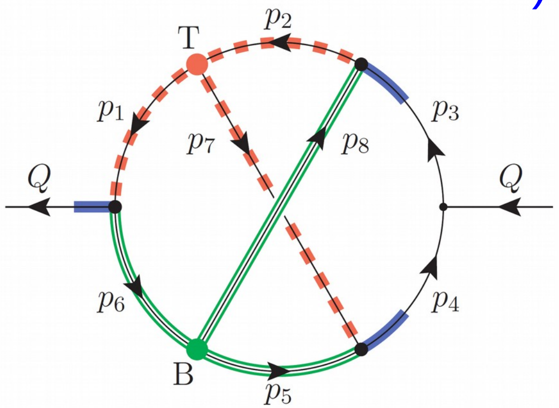
Example: 3-loop NO

- Substitute $n_9 = 0$ to the equations
- Eliminate "complicated" integrals
  (with overall $n_1 \to n_1 - 1$, the 12 IBPs are enough)




```
id Z(n1?{>=2},n2?{>=1},n3?{>=1},n4?{>=1},n5?{>=1},n6?{>=1},n7?{>=1},n8?{>=1},0) =
+Z(n1-1,n2,n3,n4-1,n5,n6,n7+1,n8,0)*rat(-n7,-1+n1)
+Z(n1-1,n2,n3,n4,n5-1,n6,n7+1,n8,0)*rat(n7,-1+n1)
+Z(n1-1,n2+1,n3-1,n4,n5,n6,n7,n8,0)*rat(-n2,-1+n1)
+Z(n1-1,n2+1,n3,n4,n5,n6,n7,n8-1,0)*rat(n2,-1+n1)
+Z(n1-1,n2,n3,n4,n5,n6,n7,n8,0)*rat(-9+2*n8+n7+2*n6+2*n5+n2+n1+4*ep,-1+n1)
+Z(n1,n2,n3,n4,n5,n6-1,n7,n8,0)
;
```

(a variant of the diamond rule)

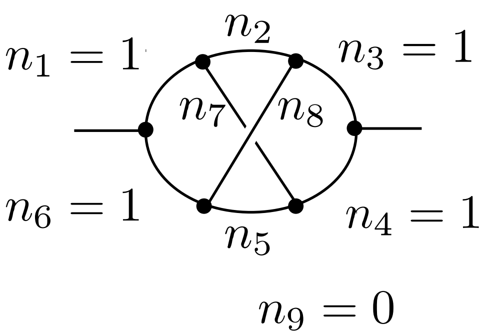


Decrease the complexity until $n_1 = 1$

The symmetry of the topology gives similar rules for n_3, n_4 and n_6

Example: 3-loop NO

- Substitute $n_1 = n_3 = n_4 = n_6 = 1$ and $n_9 = 0$ to the 120 combined equations



- Eliminate "complicated" integrals

```
id Z(1,n2?{>=2},1,1,n5?{>=1},1,n7?{>=1},n8?{>=1},0) =
+Z(1,n2-1,1,1,n5-1,1,n7,n8+1,0)*rat(-n8,-1+n2)*rat(-5+n8+2*n7+2*n5+4*ep,-3+n8+n5+n2+2*ep)
+Z(1,n2-1,1,1,n5-1,1,n7+1,n8,0)*rat(-n7,-1+n2)*rat(-7+n8+2*n7+2*n5+2*n2+4*ep,-3+n8+n5+n2+2*ep)
+Z(1,n2-1,1,1,n5,1,n7,n8,0)*rat(-1,-3+n8+n5+n2+2*ep)*rat(30-12*n8+n8^2-15*n7+3*n7*n8+2*n7^2-22*n5
    +4*n5*n8+6*n5*n7+4*n5^2-11*n2+3*n2*n8+2*n2*n7+4*n2*n5+n2^2-32*ep+6*ep*n8+8*ep*n7+12*ep*n5+6*ep*n2
    +8*ep^2,-1+n2)
+Z(1,n2,1,1,n5-1,1,n7,n8,0)*rat(2-n8-n5-2*ep,-3+n8+n5+n2+2*ep)
+Z(1,n2,1,1,n5,1,n7-1,n8,0)
+Z(1,n2,1,1,n5,1,n7,n8-1,0)*rat(7-n8-2*n7-2*n5-2*n2-4*ep,-3+n8+n5+n2+2*ep)
+Z(1,n2-1,1,0,n5,1,n7+1,n8,0)*rat(n7,-1+n2)*rat(-7+n8+2*n7+2*n5+2*n2+4*ep,-3+n8+n5+n2+2*ep)
+Z(1,n2-1,1,1,n5,0,n7,n8+1,0)*rat(n8,-1+n2)*rat(-7+n8+2*n7+2*n5+2*n2+4*ep,-3+n8+n5+n2+2*ep)
+Z(1,n2,0,1,n5,0,n7,n8+1,0)*rat(-2*n8,-3+n8+n5+n2+2*ep)
+Z(1,n2,0,1,n5,1,n7,n8,0)*rat(-9+2*n8+2*n7+3*n5+2*n2+6*ep,-3+n8+n5+n2+2*ep)
+Z(1,n2,1,1,n5,0,n7,n8,0)*rat(-9+2*n8+2*n7+2*n5+3*n2+6*ep,-3+n8+n5+n2+2*ep)
;
```

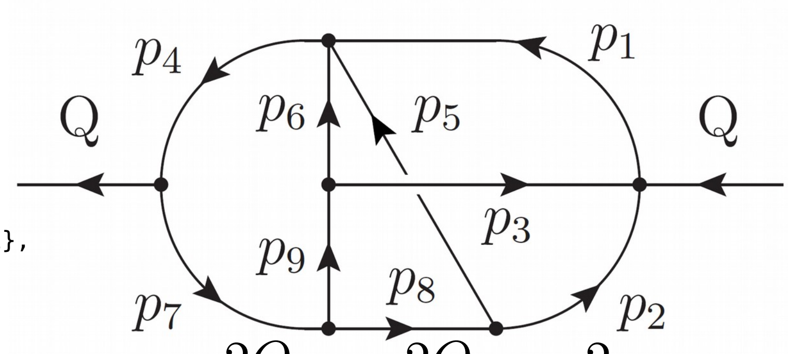
Decrease the complexity until $n_2 = 1$

Similar rules for n_5, n_7 and n_8

➡ The MI: NO(1,1,1,1,1,1,1,1,0)

Example: 4-loop BUBU

- A part of rules (reduction of n11 to 0)



$$2Q\cdot p_2,\, 2Q\cdot p_8,\, 2p_2\cdot p_3$$
$$2p_2\cdot p_9,\, 2p_3\cdot p_8$$

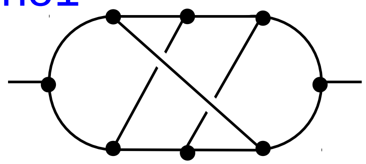
One of difficult topologies to solve, though no MI

```
id Z(n1?{>=1},n2?{>=1},n3?{>=1},n4?{>=2},n5?{>=1},n6?{>=1},n7?{>=1},n8?{>=1},n9?{>=1},
     n10?{<=0},n11?{<=-1},n12?{<=0},n13?{<=0},n14?{<=0}) =
+Z(n1,n2-1,n3,n4-1,n5+1,n6,n7,n8,n9,n10,n11+1,n12,n13,n14)*rat(n5,-1+n4)
+Z(n1,n2,n3,n4-1,n5,n6,n7+1,n8-1,n9,n10,n11+1,n12,n13,n14)*rat(-n7,-1+n4)
+Z(n1,n2,n3,n4-1,n5,n6,n7+1,n8,n9-1,n10,n11+1,n12,n13,n14)*rat(n7,-1+n4)
+Z(n1,n2,n3,n4-1,n5+1,n6,n7,n8-1,n9,n10,n11+1,n12,n13,n14)*rat(-n5,-1+n4)
+Z(n1,n2,n3,n4-1,n5,n6,n7,n8,n9,n10,n11+1,n12,n13,n14)*rat(3-2*n8-n7-n5-n14-n11-2*ep,-1+n4)
+Z(n1,n2,n3,n4,n5,n6,n7-1,n8,n9,n10,n11+1,n12,n13,n14)*rat(-1,1)
+Z(n1,n2,n3,n4,n5,n6,n7,n8-1,n9,n10,n11+1,n12,n13,n14)*rat(-1,1)
+Z(n1,n2,n3,n4,n5,n6,n7,n8,n9-1,n10,n11+1,n12,n13,n14)
;
id Z(n1?{>=1},n2?{>=1},n3?{>=1},1,n5?{>=1},n6?{>=1},n7?{>=1},n8?{>=1},n9?{>=1},
     n10?{<=0},n11{<=-1},n12?{<=0},n13?{<=0},n14?{<=0}) =
+Z(n1,n2,n3,1,n5,n6,n7,n8,n9,n10,n11+1,n12+1,n13-1,n14)*rat(n12,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3,1,n5,n6,n7,n8,n9,n10+1,n11+1,n12,n13-1,n14)*rat(-n10,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3,1,n5,n6,n7+1,n8-1,n9,n10,n11+1,n12,n13,n14)*rat(-n7,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3,1,n5,n6,n7+1,n8,n9-1,n10,n11+1,n12,n13,n14)*rat(n7,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3+1,1,n5,n6-1,n7,n8,n9,n10,n11+1,n12,n13,n14)*rat(-n3,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3+1,1,n5,n6,n7,n8,n9-1,n10,n11+1,n12,n13,n14)*rat(n3,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3,1,n5,n6,n7,n8,n9,n10,n11+1,n12,n13,n14)*rat(4+n9-n8-n6-n5-n2-n14-n12-n11-n10-n1-3*ep,
    -8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3,1,n5,n6,n7-1,n8,n9,n10,n11+1,n12,n13,n14)*rat(-1,1)
+Z(n1,n2,n3,1,n5,n6,n7-1,n8,n9,n10,n11+1,n12,n13,n14+1)*rat(n14,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3,1,n5,n6,n7-1,n8,n9,n10,n11+2,n12,n13,n14)*rat(-1-n11,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3,1,n5,n6,n7,n8-1,n9,n10,n11+1,n12,n13,n14+1)*rat(-n14,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3,1,n5,n6,n7,n8-1,n9,n10,n11+2,n12,n13,n14)*rat(1+n11,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3,1,n5,n6,n7,n8,n9-1,n10,n11+1,n12,n13,n14+1)*rat(-n14,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3,1,n5,n6,n7,n8,n9-1,n10,n11+2,n12,n13,n14)*rat(1+n11,-8+n9+n8+n7+n6+n5+n3+n2+n14+n13+n12+n11+n10+n1+5*ep)
+Z(n1,n2,n3,0,n5,n6,n7,n8,n9,n10,n11+1,n12,n13,n14)
;
```

3274 lines for whole BUBU reduction

Run Forcer: result looks like...

no1

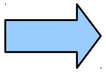


one of the top-level topology

11 propagators

3 irreducible numerators

How to convert Qgraf output to Forcer input?



Jos's talk

```
[no1(2,2,2,2,2,2,2,2,2,2,2,-1,-1,-1)] =

-10/9*num(1+2*ep)^2*num(2+5*ep)*num(3+2*ep)^2*num(3+5*ep)*num(4+5*ep)*num(6+5*ep)*num(7+5*ep)*num(8+5*ep
)*num(9+5*ep)*num(36141384+167650024*ep+369157793*ep^2+504389598*ep^3+470560515*ep^4+312347786*ep^5+1497
70838*ep^6+51734214*ep^7+12600912*ep^8+2060632*ep^9+203648*ep^10+9216*ep^11)*den(1+ep)^2*den(2+ep)^2*den
(2+3*ep)*den(3+ep)^2*den(4+3*ep)*den(5+3*ep)*den(7+3*ep)*den(8+3*ep)*Master(no1)

+2/9*num(1+2*ep)*num(1+4*ep)^2*num(3+2*ep)*num(39337381531008+422506983834144*ep+2126828256064272*ep^2+6
682923999248124*ep^3+14715479582570820*ep^4+24142592323497543*ep^5+30606097414180459*ep^6+30663886432898
386*ep^7+24613240685251374*ep^8+15944111435166437*ep^9+8353498138352567*ep^10+3530865125153640*ep^11+119
5052526807120*ep^12+319518770241334*ep^13+66015276933500*ep^14+10173244132808*ep^15+1101566031376*ep^16+
74820000000*ep^17+2400000000*ep^18)*den(1+ep)^2*den(2+ep)^2*den(2+3*ep)*den(3+ep)^2*den(4+3*ep)*den(5+3*
ep)*den(7+3*ep)*den(8+3*ep)*Master(no6)

-4/9*num(ep)^2*num(1+2*ep)*num(3+2*ep)*num(4970268890523930816+65981799142896214872*ep+41636699974387213
0470*ep^2+1663841711414114631648*ep^3+4730696245772216998455*ep^4+10190116418687033776180*ep^5+172831512
25628577356718*ep^6+23674869565869007631757*ep^7+26649665048245779930527*ep^8+24945440138420091798098*ep
^9+19571616320498135318722*ep^10+12933081950816489701881*ep^11+7214358008133792648788*ep^12+339681000556
8803286172*ep^13+1346680080420400500352*ep^14+447318716827968227680*ep^15+123500240869574621248*ep^16+28
008584917867939712*ep^17+5129371482425778688*ep^18+739826312414941184*ep^19+80910145880457216*ep^20+6306
386119458816*ep^21+312125440000000*ep^22+7372800000000*ep^23)*den(1+ep)^2*den(2+ep)^2*den(3+ep)^2*den(3+
4*ep)*den(4+3*ep)*den(5+3*ep)*den(5+4*ep)*den(7+3*ep)*den(7+4*ep)*den(8+3*ep)*den(9+4*ep)*den(11+4*ep)*d
en(13+4*ep)*Master(lala)
```

(cont'd on next page)

```
+ ( other 17 terms )
```

(cont'd)

```
+1/69984*num(-1+2*ep)^3*num(72419162019449762165096448000000+3199339702731261014302808309760000*ep+6756458
7046942832244281149906944000*ep^2+91350459184391655670944774398730240*ep^3+891289886775817303600596179144
718336*ep^4+66930375969349547572861052984235786624*ep^5+40235415927130336283161325668026920448*ep^6+19871
3617144519275979305008924953882368*ep^7+82091697044731460372708583674670235468 8*ep^8+2873881635171369729
2804388788688381310 08*ep^9+86079527794109515224134940943137287932 80*ep^10+2221477733684447301324246670 01
94857424432*ep^11+4963564313550399839873313962385333772039 2*ep^12+9627197032010951955470647409501905888 4
972*ep^13+1620920042433367369484228940869152451887 10*ep^14+2360110255144947383954778523265243802462 08*ep
^15+29411662512212795213699668738158804546689 7*ep^16+30622594561109334098894115787780494656562 2*ep^17+25
0395537138862785223188155623245534964532*ep^18+1278490968230930766213197239347627354187 74*ep^19-32311698
6586416421659980377524004225109 97*ep^20-1839323952847706865576065109996382637244 34*ep^21-284795600836347
4571865784913534305788703 26*ep^22-3150229191924838122866754344782081602769 10*ep^23-2823055800376111021 51
35831194998484916316 4*ep^24-2129361990228716919075740753830553714327 06*ep^25-1364459935186578669634780 40
015820540166838*ep^26-734900782788793457862107578060975188670 18*ep^27-318984658302192402132855983340890 3
5466304*ep^28-965138248495143989347275710922069398441 4*ep^29-42123656180630018160833079911348461388 4*ep^
30+2017002468644720352265271521884033262822*ep^31+183287476156286668669830509056864470289 9*ep^32+1096707
581456919232171368580075436795092*ep^33+524523702662773366611140017451441153706*ep^34+212149752741671710
6493850325590950347 96*ep^35+742509010290102120707359886108873337 49*ep^36+2272051252243184038118975458520
5484412*ep^37+61032923996908799039999592331232436 84*ep^38+14397363931112773511749107695815768 00*ep^39+29
75285303229757980968392200467472 48*ep^40+536061820919166327089727654950660 48*ep^41+835965908902794893568
4267151014592*ep^42+1117001205642226519475032390006784*ep^43+126129308732809465322496308824064*ep^44+118
1003341832521121415433402777 6*ep^45+8928238489352988841796013096 96*ep^46+52384336126816611070473928704*e
p^47+2238933537865339140484104192*ep^48+620290211369818521600000000*ep^49+83626417685790720000000 0*ep^50)
*den(ep)^6*den(-1+ep)*den(1+ep)^6*den(1+2*ep)^2*den(1+3*ep)^2*den(-2+ep)*den(2+ep)^6*den(2+3*ep)^2*den(3
+ep)^3*den(3+2*ep)^2*den(3+4*ep)*den(4+3*ep)^2*den(5+2*ep)^2*den(5+3*ep)^2*den(5+4*ep)*den(7+3*ep)^2*den
(7+4*ep)*den(8+3*ep)*den(9+4*ep)*den(11+4*ep)*den(13+4*ep)*G10*G20*G30
```

4.2 hours on a desktop PC with “tform -w4”

Exact in ϵ Optionally: ϵ -expansions in intermediate steps

Checks

- Recomputing known results: strong non-trivial checks
  - Reproduced the 4-loop QCD β-function [Ritbergen, Vermaseren, Larin '97; Czakon '04]
  - Checked the gauge invariance, all powers of ξ

    Forcer gives 4-loop propagators and vertices*, exact in ϵ, all powers of ξ

    * with nullifying external momenta except two of them
  - Also with using background field method

with no gauge parameter	10 minutes
with one power	38 minutes
with FULL gauge parameters	8.5 hours

on a decent 24 core machine (6 "tform -w4" jobs)

  - Reproduced low-N NS splitting functions ⇨ Andreas's talk

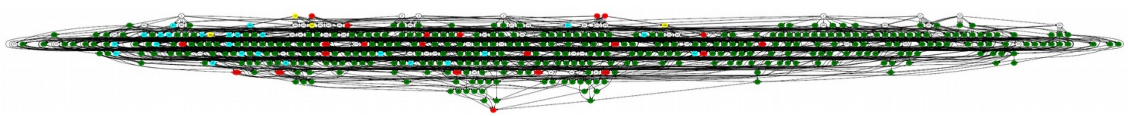
    GLS, N=2,3,4 [Baikov, Chetyrkin, Kühn; Velizhanin]

Summary

Summary

- Forcer: a “4-loop extension” of Mincer
  for massless propagator-type Feynman integrals

  - Highly complicated structure of the program / equations
    ⇒ Automatization: write a program for generating a code

    for 

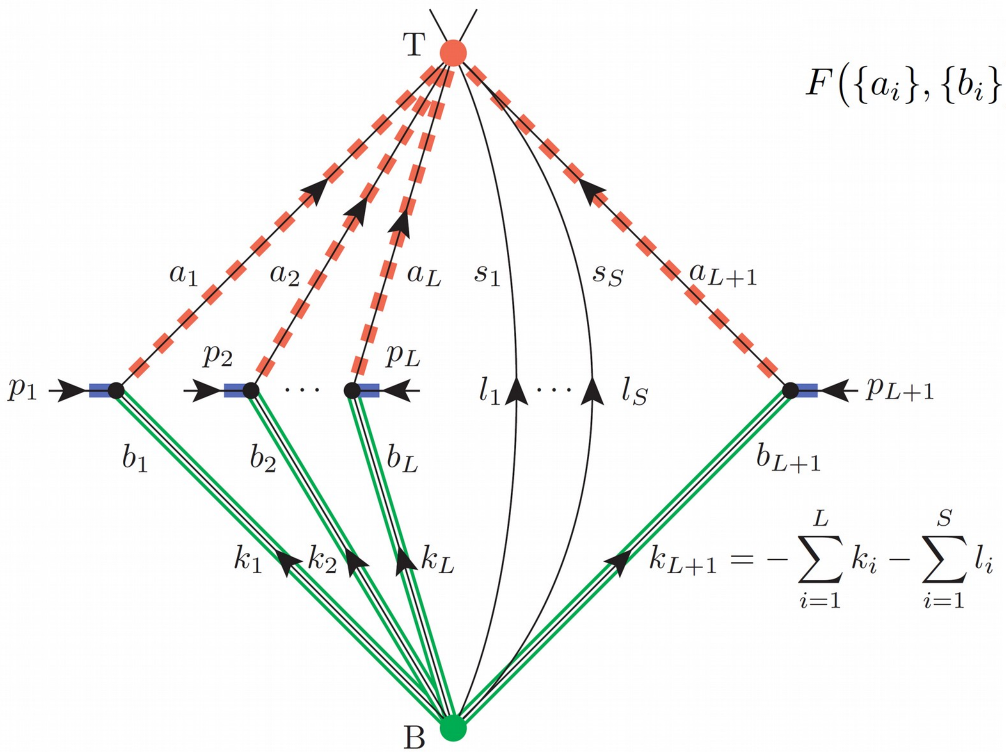
    Manual rules are derived with the aid of computers

- Correctness of the program was checked for known results

Backup

Diamond rule

[Ruijl, TU, Vermaseren ’15]



$$F(\{a_i\},\{b_i\}) = \left[\prod_{i=1}^{L}\int d^D k_i\right]\left[\prod_{i=1}^{S}\int d^D l_i\right] \times \left[\prod_{i=1}^{L+1}\frac{k_i^{\mu_1^{(i)}}\dots k_i^{\mu_{N_i}^{(i)}}}{\left[(k_i+p_i)^2+m_i^2\right]^{a_i}(k_i^2)^{b_i}}\right]\left[\prod_{i=1}^{S}\frac{l_i^{\nu_1^{(i)}}\dots l_i^{\nu_{R_i}^{(i)}}}{(l_i^2)^{s_i}}\right]$$

$$(L+S)D+\sum_{i=1}^{L+1}(N_i-a_i-2b_i)+\sum_{i=1}^{S}(R_i-2s_i) = \sum_{i=1}^{L+1} a_i \boldsymbol{A}_i^+\left[\boldsymbol{B}_i^- -(p_i^2+m_i^2)\right]$$

Diamond rule

- Explicit summation formula

$$F(\{a_i\},\{b_i\},\{c_i\}) =$$

$$\sum_{r=1}^{L+1}\Bigg[\Bigg(\prod_{\substack{i=1\\i\neq r}}^{L+1}\sum_{k_i^+=0}^{b_i-1}\Bigg)\Bigg(\prod_{i=1}^{L+1}\sum_{k_i^-=0}^{c_i-1}\Bigg)(-1)^{k^-}\frac{k_r^+(k^+ + k^- - 1)!}{\prod_{i=1}^{L+1}k_i^+!k_i^-!}(E+k^+)_{-k^+-k^-}$$

$$\times\Bigg(\prod_{i=1}^{L+1}(a_i)_{k_i^+ + k_i^-}\Bigg)F\left(\{a_i+k_i^+ + k_i^-\},\{b_i-k_i^+\},\{c_i-k_i^-\}\right)\Bigg]_{k_r^+=b_r}$$

$$+\sum_{r=1}^{L+1}\Bigg[\Bigg(\prod_{i=1}^{L+1}\sum_{k_i^+=0}^{b_i-1}\Bigg)\Bigg(\prod_{\substack{i=1\\i\neq r}}^{L+1}\sum_{k_i^-=0}^{c_i-1}\Bigg)(-1)^{k^-}\frac{k_r^-(k^+ + k^- - 1)!}{\prod_{i=1}^{L+1}k_i^+!k_i^-!}(E+k^+ + 1)_{-k^+-k^-}$$

$$\times\Bigg(\prod_{i=1}^{L+1}(a_i)_{k_i^+ + k_i^-}\Bigg)F\left(\{a_i+k_i^+ + k_i^-\},\{b_i-k_i^+\},\{c_i-k_i^-\}\right)\Bigg]_{k_r^-=c_r}$$

$$E=(L+S)D+\sum_{i=1}^{L+1}(N_i-a_i-2b_i)+\sum_{i=1}^{S}(R_i-2s_i)$$

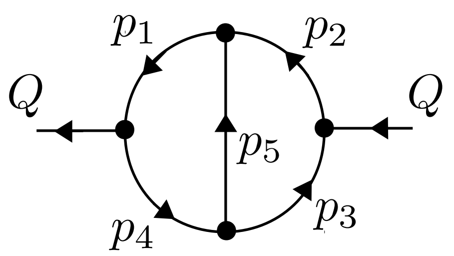
$$k^+=\sum_{i=1}^{L+1}k_i^+ \qquad k^-=\sum_{i=1}^{L+1}k_i^-$$

Avoids spurious poles

Triangle/diamond rule may be not the best

- The triangle rule does not change the total complexity of integrals

```
id Z(n1?{>=1},n2?{>=1},n3?{>=1},n4?{>=1},n5?{>=1}) =
+Z(n1,n2,n3-1,n4+1,n5)*rat(n4,-4+2*n5+n4+n1+2*ep)
+Z(n1,n2,n3,n4+1,n5-1)*rat(-n4,-4+2*n5+n4+n1+2*ep)
+Z(n1+1,n2-1,n3,n4,n5)*rat(n1,-4+2*n5+n4+n1+2*ep)
+Z(n1+1,n2,n3,n4,n5-1)*rat(-n1,-4+2*n5+n4+n1+2*ep)
;
```



- The following rule (a solution of 6+30 equations) decreases the complexity and gives only 3 terms in the RHS

```
id Z(n1?{>=1},n2?{>=1},n3?{>=1},n4?{>=1},n5?{>=1}) =
+Z(n1,n2-1,n3,n4,n5)*rat(-7+2*n5+n4+n3+n2+n1+3*ep,-3+n5+n2+n1+ep)
+Z(n1,n2,n3-1,n4,n5)*rat(-7+2*n5+n4+n3+n2+n1+3*ep,-3+n5+n4+n3+ep)
+Z(n1,n2,n3,n4,n5-1)*rat(3-n5-n4-n1-ep,-3+n5+n2+n1+ep)
    *rat(-4+n5+n4+n1+2*ep,-3+n5+n4+n3+ep)
;
```

- Possibility of more optimizations??