New developments in FORM

Takahiro Ueda

Nikhef Theory Group

FORM website: `https://www.nikhef.nl/~form/`
Source code: `https://github.com/vermaseren/form`

B. Ruijl, TU, J.A.M. Vermaseren, *FORM version 4.2*, arXiv:1707.06453

Nikhef

August 21, 2017
University of Washington, Seattle
ACAT 2017

Outline

Introduction: background

FORM: design principles

New release: FORM 4.2

Form is

Form is

a toolkit for formula manipulation

Form is

a toolkit for formula manipulation

Symbolic computation instead of numerical computation

$$A = \pi r^2$$

vs.

$$A = 3.141593 \times 5.000000 \times 5.000000$$
$$= 78.53982$$

Form is

a toolkit for formula manipulation

Symbolic computation instead of numerical computation

$$A = \pi r^2$$

vs.

$$A = 3.141593 \times 5.000000 \times 5.000000$$
$$= 78.53982$$

Often used in theoretical particle physics though not restricted to any specific field

Why symbolic?

Why symbolic?

- Exact

Why symbolic?

Exact

More information about
the mathematical structure

$$B := A^n = \pi^n r^{2n}$$

vs.

$$B = (78.53982)^{10} = 8.930978 \times 10^{18}$$

- 👉 *but, see Laporta's talk*
  PSLQ to recover analytical result

Why symbolic?

Exact

More information about
the mathematical structure

$$B := A^n = \pi^n r^{2n}$$

vs.

$$B = (78.53982)^{10} = 8.930978 \times 10^{18}$$

👉 *but, see Laporta's talk*
PSLQ to recover analytical result

Theory is described in math (symbolic)

Theories in QFTs

Framework of Quantum Field Theory

Theories in QFTs

Framework of Quantum Field Theory

Theory $=$ Lagrangian $\mathcal{L}$

(model)

Theories in QFTs

Quantum electrodynamics (QED)

$$\mathcal{L} = -\frac{1}{4} F_{\mu\nu} F^{\mu\nu} \quad + \bar{\psi}(i\not{D} - m)\psi$$

photon (light) electron (matter)

Theories in QFTs

Quantum electrodynamics (QED)

interaction

$$\mathcal{L} = -\frac{1}{4} F_{\mu\nu} F^{\mu\nu} \quad + \bar{\psi}(i\not{D} - m)\psi$$

photon (light) electron (matter)

Theories in QFTs

Quantum chromodynamics (QCD)

$$\mathcal{L} = -\frac{1}{4} F^{a}_{\mu\nu} F^{\mu\nu,a} + \bar{\psi}(i\not{D} - m)\psi$$

gluon quark

Theories in QFTs

Quantum chromodynamics (QCD)

gluon self-interaction

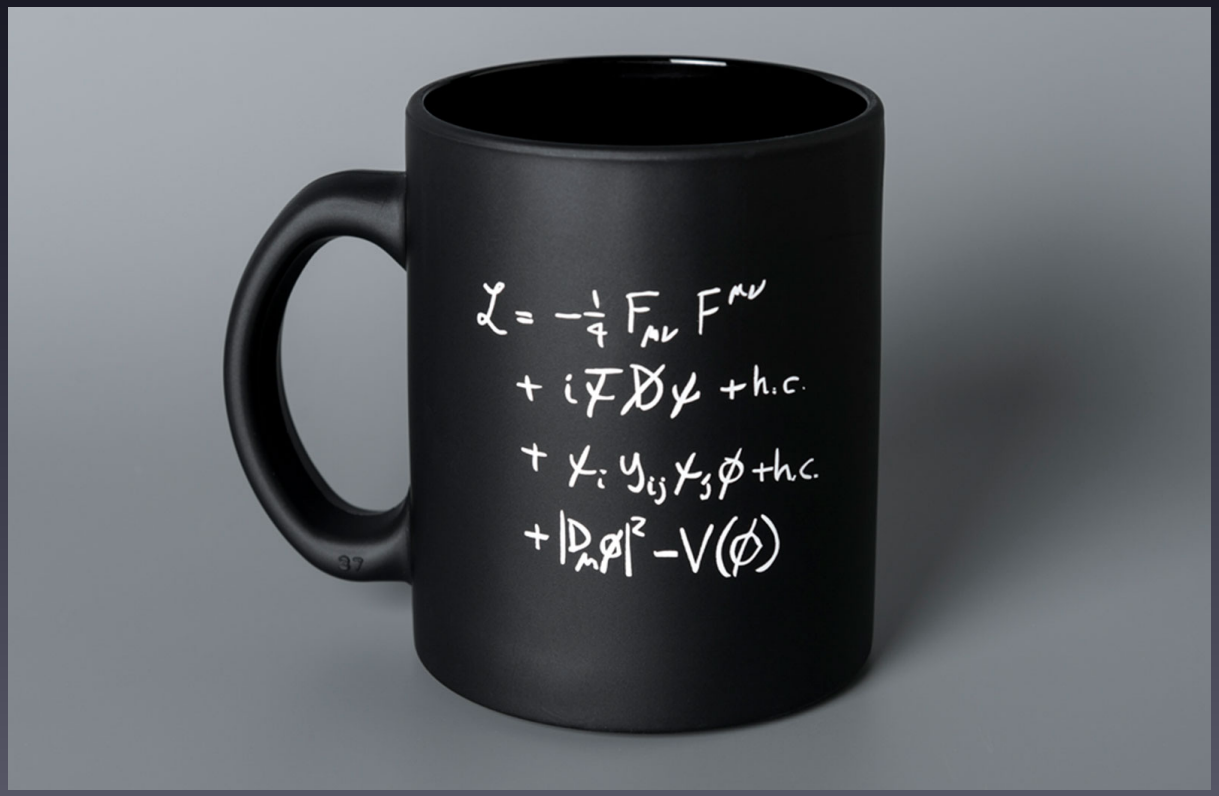
quark-gluon interaction

$$\mathcal{L} = -\frac{1}{4} F^{a}_{\mu\nu} F^{\mu\nu,a} + \bar{\psi}(i\not{D} - m)\psi$$

gluon quark + Faddeev–Popov ghost

Theories in QFTs

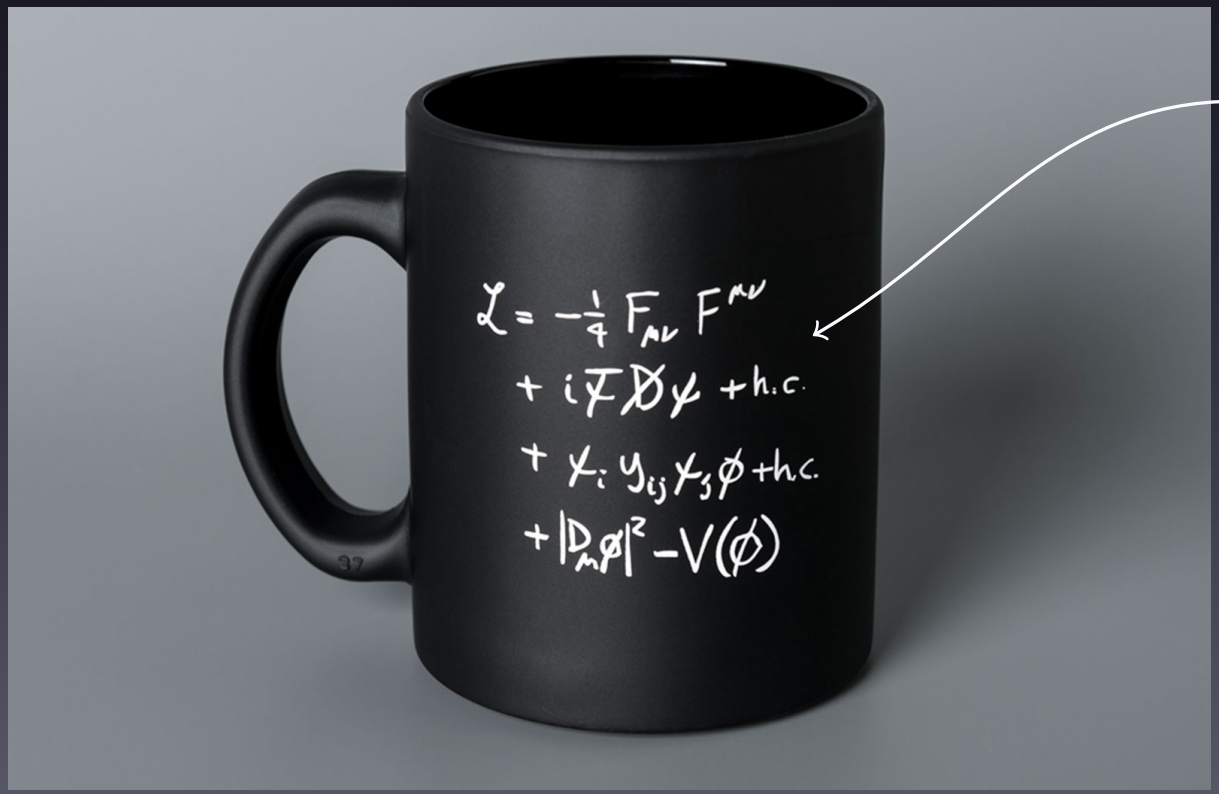
The Standard Model (SM)



Picture from CERN Shop

Theories in QFTs

The Standard Model (SM)

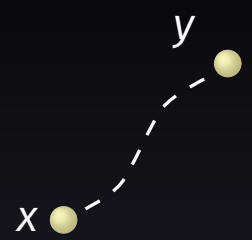


quarks & leptons
a Higgs boson
QED + QCD
+ weak interaction

Picture from CERN Shop

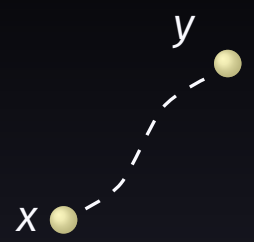
Theory prediction

Theory prediction



electron
quantum effect?

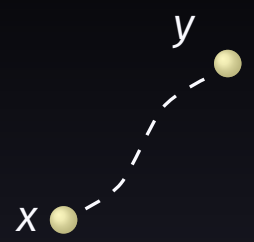
Theory prediction



electron
quantum effect?

$$= \mathcal{N} \left\langle 0 \middle| T\left\{ \psi(x) \bar{\psi}(y) \exp\left[i \int d^4 z \, \mathcal{L}_{\text{int}}(z) \right] \right\} \middle| 0 \right\rangle$$

Theory prediction

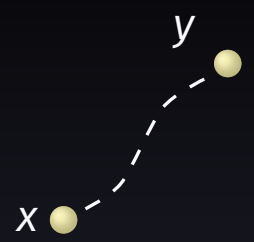


electron
quantum effect?

Your Theory

$$= \mathcal{N}\left\langle 0 \middle| T\left\{ \psi(x)\bar{\psi}(y) \exp\left[i \int d^4z \, \mathcal{L}_{\text{int}}(z) \right] \right\} \middle| 0 \right\rangle$$

Theory prediction



electron
quantum effect?

Your Theory

$$= \mathcal{N} \left\langle 0 \left| T \left\{ \psi(x) \bar{\psi}(y) \exp \left[i \int d^4 z \, \mathcal{L}_{\text{int}}(z) \right] \right\} \right| 0 \right\rangle$$

perturbative expansion

$$= \square + \alpha \square + \alpha^2 \square + \alpha^3 \square + \ldots$$

- *but, see Kronfeld's talk non-perturbative method*

QED coupling: $\alpha \sim \frac{1}{137} \ll 1$

Perturbative expansion: How to?

Perturbative expansion: How to?

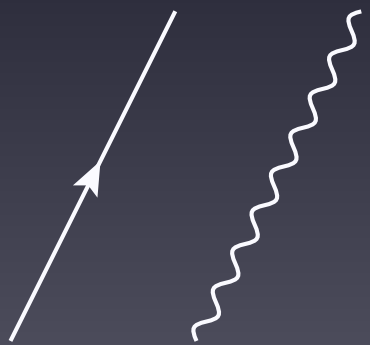
Picture from Nobelprize.org

Draw
Feynman diagrams

Perturbative expansion: How to?

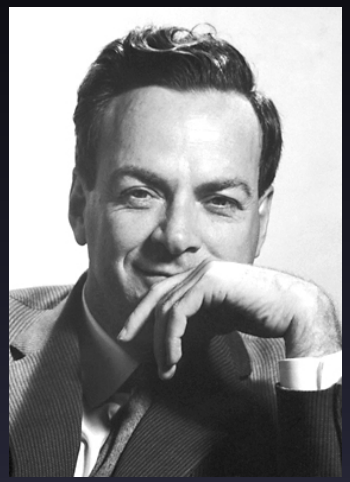
Picture from Nobelprize.org

Draw
Feynman diagrams



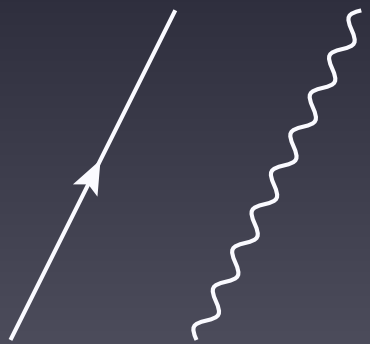
freely propagating
electron and photon

Perturbative expansion: How to?

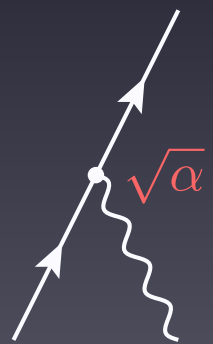


Picture from Nobelprize.org

Draw
Feynman diagrams

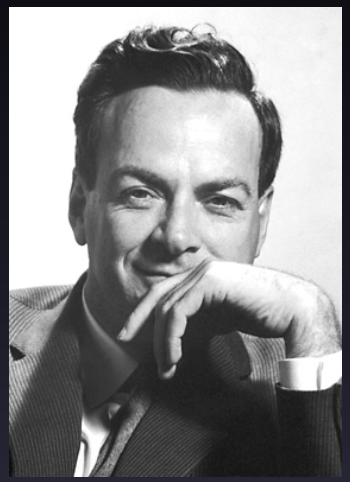


freely propagating
electron and photon



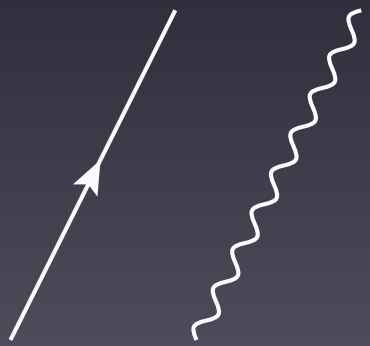
interaction
vertex

Perturbative expansion: How to?

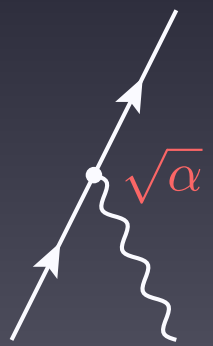


Picture from Nobelprize.org

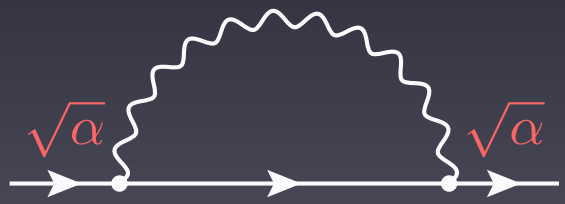
Draw
Feynman diagrams



freely propagating
electron and photon

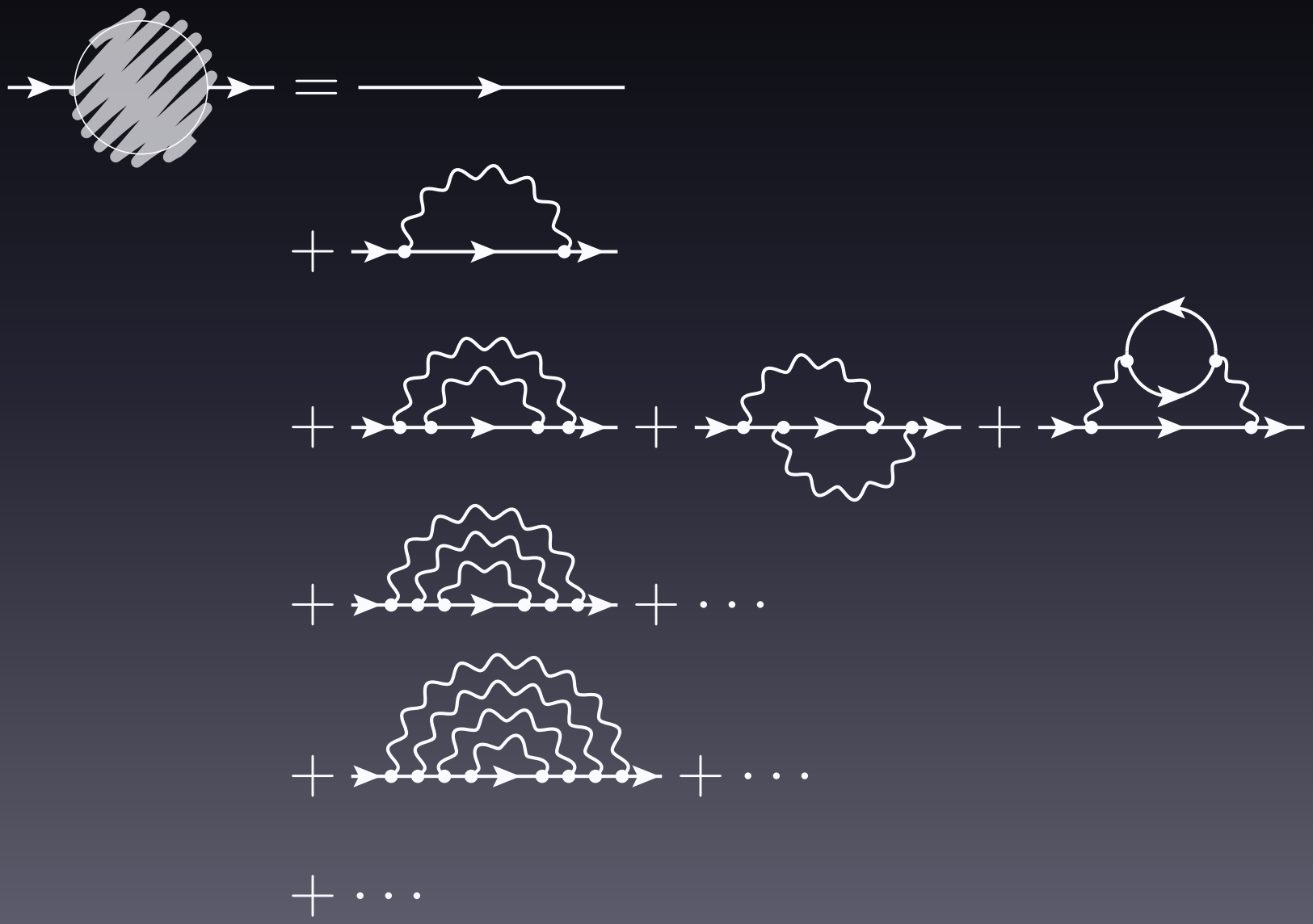


interaction
vertex

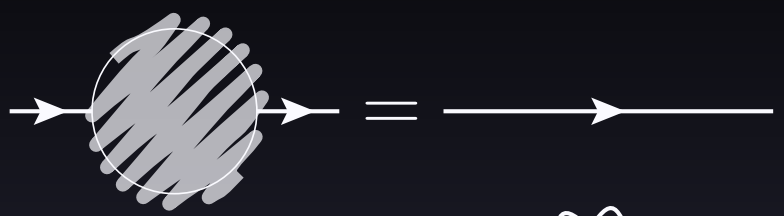


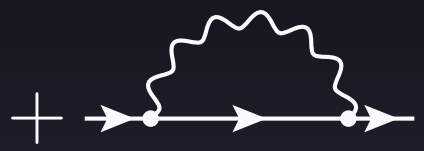
$\mathcal{O}(\alpha)$-contribution

Perturbative expansion: How to?

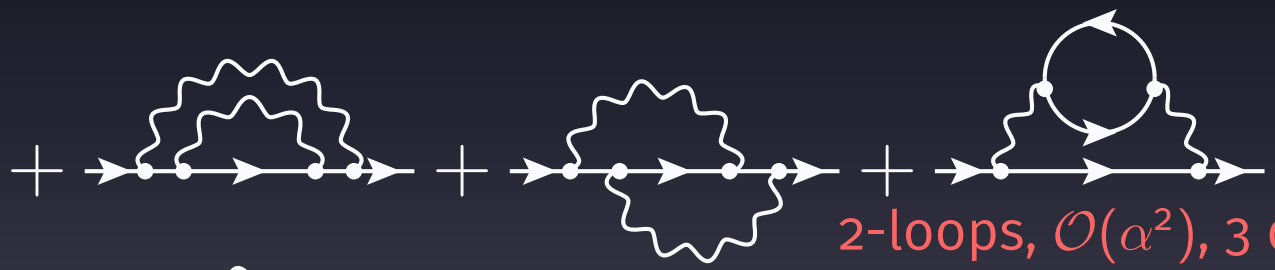


Perturbative expansion: How to?

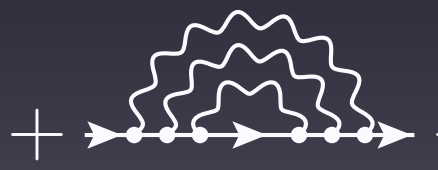




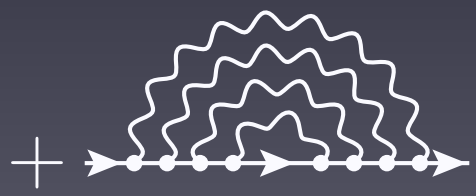
1-loop, $\mathcal{O}(\alpha)$, 1 diagram



2-loops, $\mathcal{O}(\alpha^2)$, 3 diagrams



3-loops, $\mathcal{O}(\alpha^3)$, 18 diagrams



4-loops, $\mathcal{O}(\alpha^4)$, 153 diagrams

$+ \cdots$ Translated into math expressions

Life is not easy

Life is not easy

Many diagrams, possibly $\sim$ 1M

Life is not easy

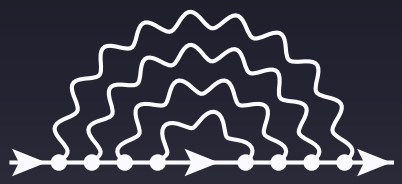
Many diagrams, possibly $\sim$ 1M

Complicated structure

Life is not easy

Many diagrams, possibly $\sim$ 1M

Complicated structure

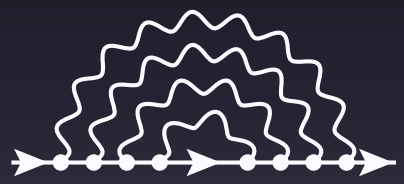


$$\rightarrow \mathrm{Tr}(\gamma^{\mu_1}\gamma^{\mu_2} \ldots \gamma^{\mu_{16}})$$
$$= 4g^{\mu_1\mu_2}g^{\mu_3\mu_4} \ldots g^{\mu_{15}\mu_{16}} + \cdots$$
$$\frac{16!}{2^{16/2}(16/2)!} = 2027025 \text{ terms in total}$$

Life is not easy

Many diagrams, possibly $\sim$ 1M

Complicated structure



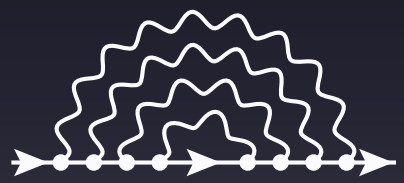
$$\rightarrow \mathrm{Tr}(\gamma^{\mu_1}\gamma^{\mu_2} \ldots \gamma^{\mu_{16}})$$
$$= 4g^{\mu_1\mu_2}g^{\mu_3\mu_4} \ldots g^{\mu_{15}\mu_{16}} + \cdots$$
$$\frac{16!}{2^{16/2}(16/2)!} = 2027025 \text{ terms in total}$$

Divergent integral

Life is not easy

Many diagrams, possibly $\sim$ 1M

Complicated structure



$$\to \mathrm{Tr}(\gamma^{\mu_1}\gamma^{\mu_2} \ldots \gamma^{\mu_{16}})$$
$$= 4g^{\mu_1\mu_2}g^{\mu_3\mu_4} \ldots g^{\mu_{15}\mu_{16}} + \cdots$$
$$\frac{16!}{2^{16/2}(16/2)!} = 2027025 \text{ terms in total}$$

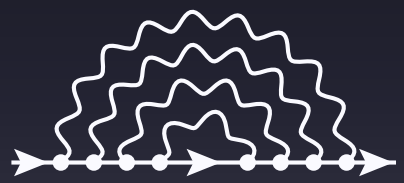
Divergent integral

$\sim \infty$

Life is not easy

Many diagrams, possibly $\sim$ 1M

Complicated structure



$$\to \mathrm{Tr}(\gamma^{\mu_1}\gamma^{\mu_2}\ldots\gamma^{\mu_{16}})$$
$$= 4g^{\mu_1\mu_2}g^{\mu_3\mu_4}\ldots g^{\mu_{15}\mu_{16}} + \cdots$$
$$\frac{16!}{2^{16/2}(16/2)!} = 2027025 \text{ terms in total}$$

Divergent integral

$\sim \infty$

Keep the divergence as
a symbol until cancelled

Summary so far

Summary so far

Perturbative QFT requires
symbolic manipulation with many terms

Summary so far

Perturbative QFT requires
symbolic manipulation with many terms

Basic algebraic operations. No need for fancy automatic integration or knowledge of special functions

Summary so far

Perturbative QFT requires
symbolic manipulation with many terms

Basic algebraic operations. No need for
fancy automatic integration or knowledge of
special functions
Huge expression size $\sim$ 1.5–2TB

Summary so far

Perturbative QFT requires
symbolic manipulation with many terms

Basic algebraic operations. No need for fancy automatic integration or knowledge of special functions
Huge expression size $\sim$ 1.5–2TB

Q. Which computer algebra system(s) can be suited?

   FORM GiNaC Maple Mathematica Maxima Reduce SageMath ...

Summary so far

Perturbative QFT requires
symbolic manipulation with many terms

Basic algebraic operations. No need for fancy automatic integration or knowledge of special functions
Huge expression size $\sim$ 1.5–2TB

Q. Which computer algebra system(s) can be suited?

   FORM GiNaC Maple Mathematica Maxima Reduce SageMath ...

Outline

Expression as tree

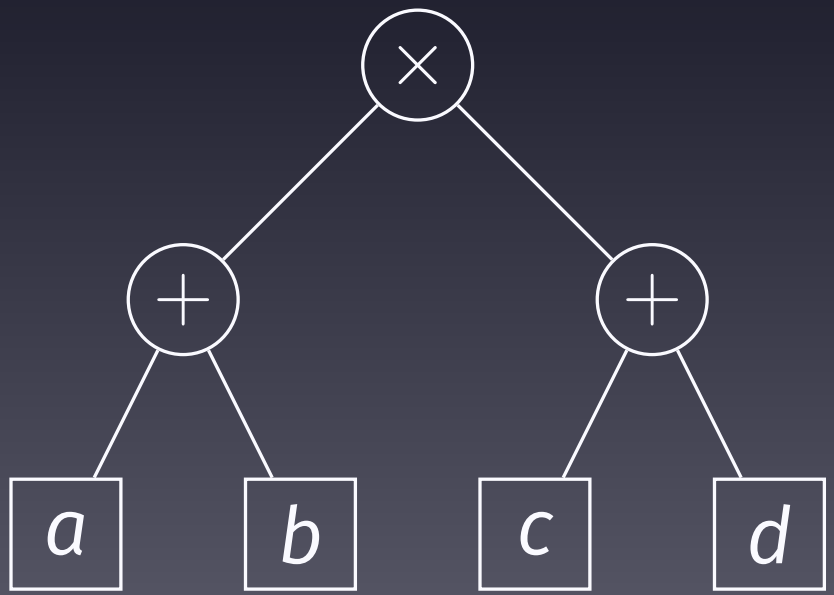
Most of computer algebra systems

$$(a + b)(c + d)$$

Expression as tree

Most of computer algebra systems

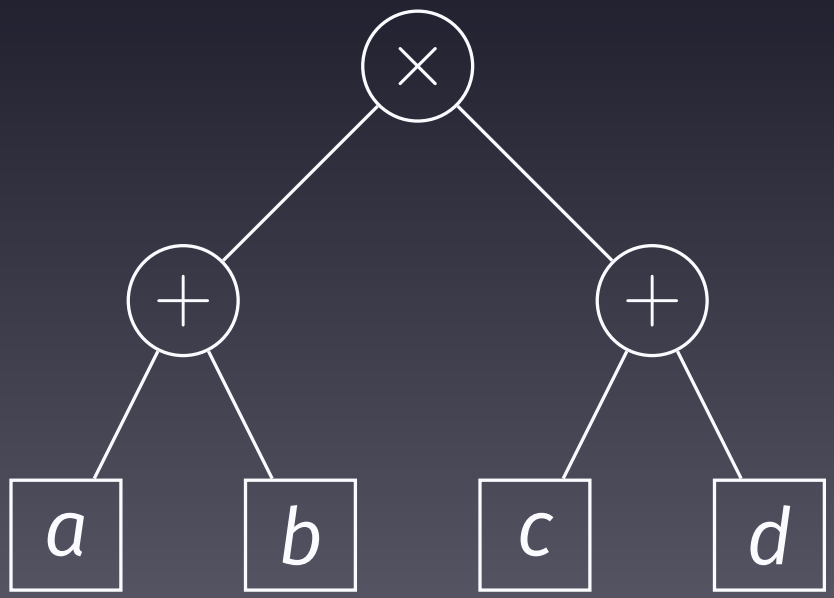
$$(a + b)(c + d)$$



Expression as tree

Most of computer algebra systems

$(a + b)(c + d)$

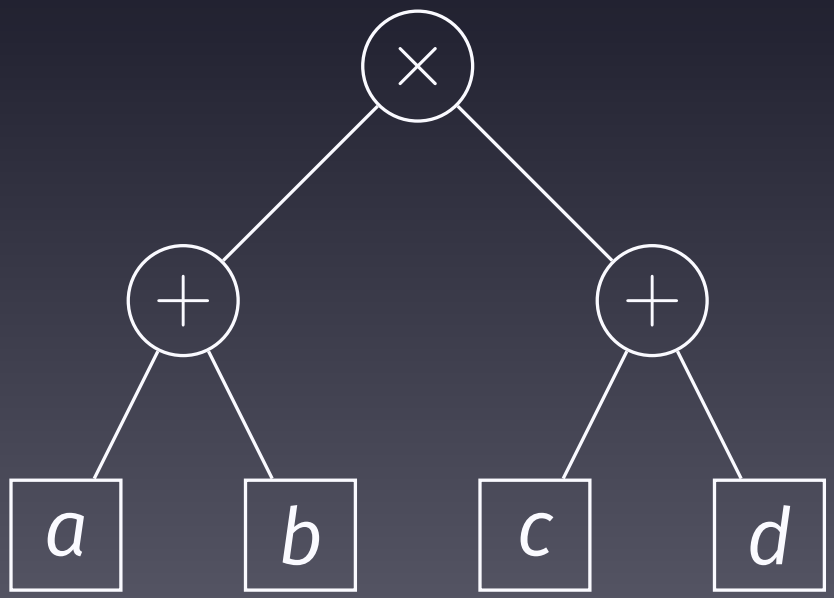


Elegant
Recursive structure ☺

Expression as tree

Most of computer algebra systems

$(a + b)(c + d)$



Elegant
Recursive structure ☺

Random access may cause
heavy swapping ☹

Expression as sequence

Form

$$(a + b)(c + d)$$

Expression as sequence

FORM always expands expressions

$$(a + b)(c + d) = ac + ad + bc + bd$$

Expression as sequence

FORM always expands expressions

$$(a+b)(c+d) = ac + ad + bc + bd$$

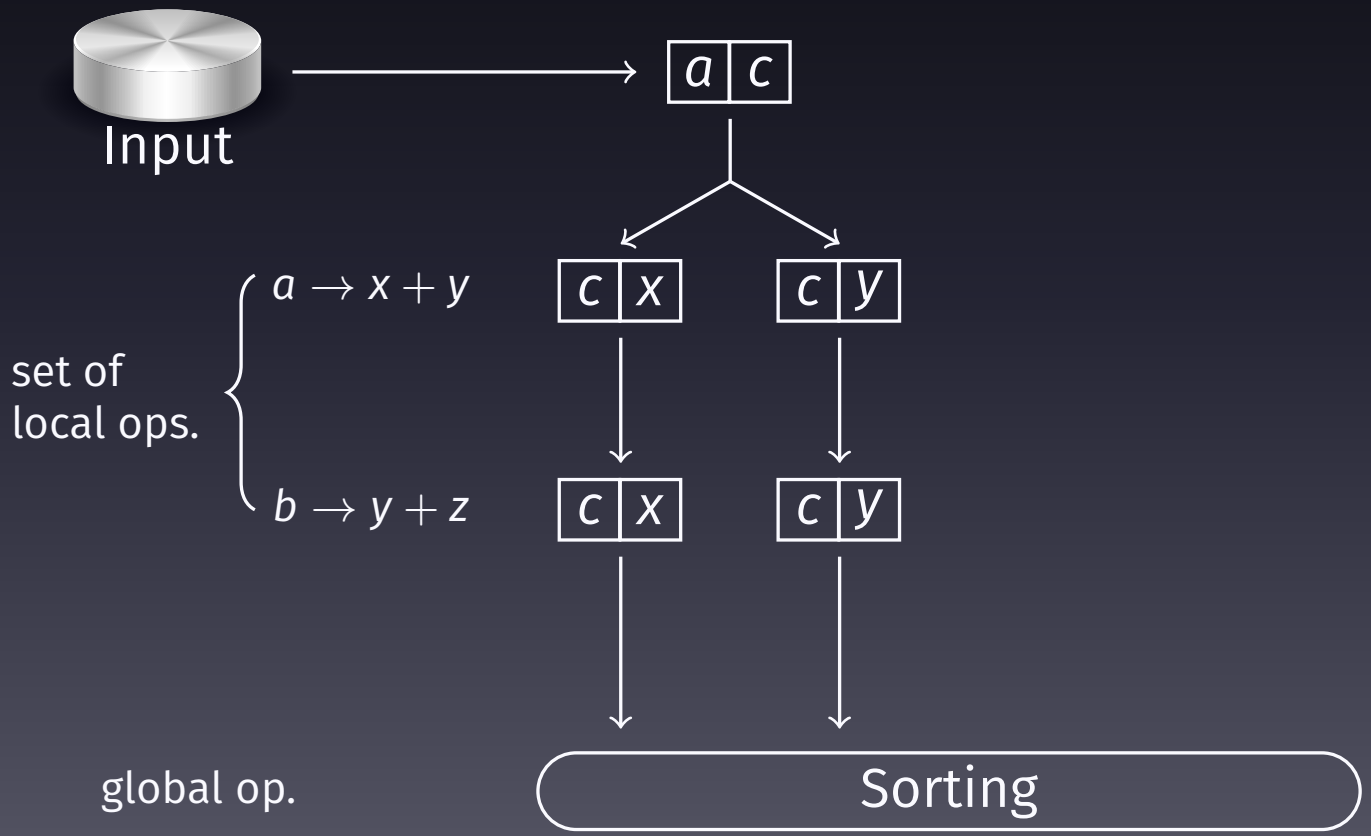
Sequence of (sorted) terms
Each term has complete information

Local & global operations

Local operations on each term
Global operations, e.g., sorting

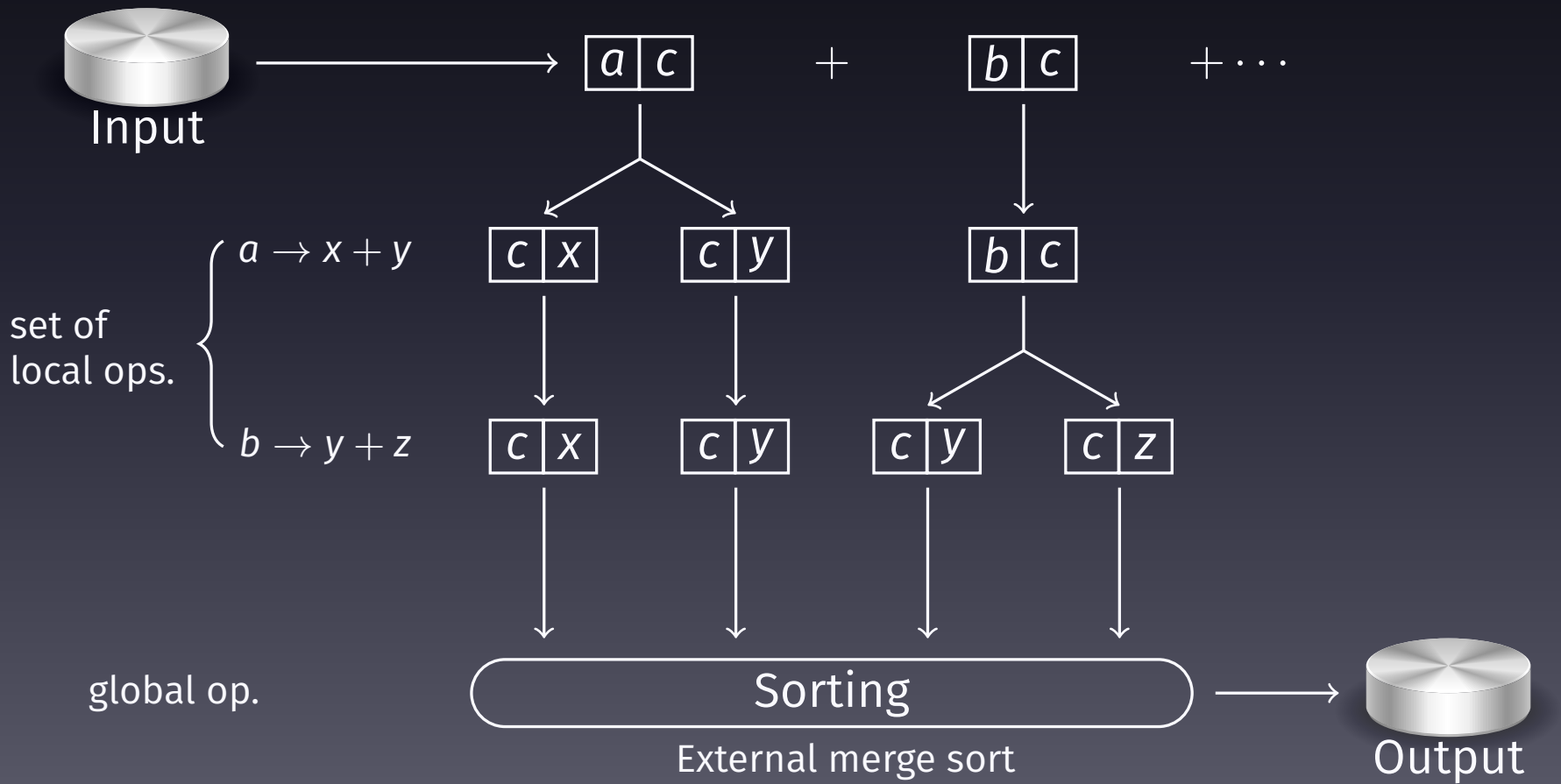
Local & global operations

Local operations on each term
Global operations, e.g., sorting



Local & global operations

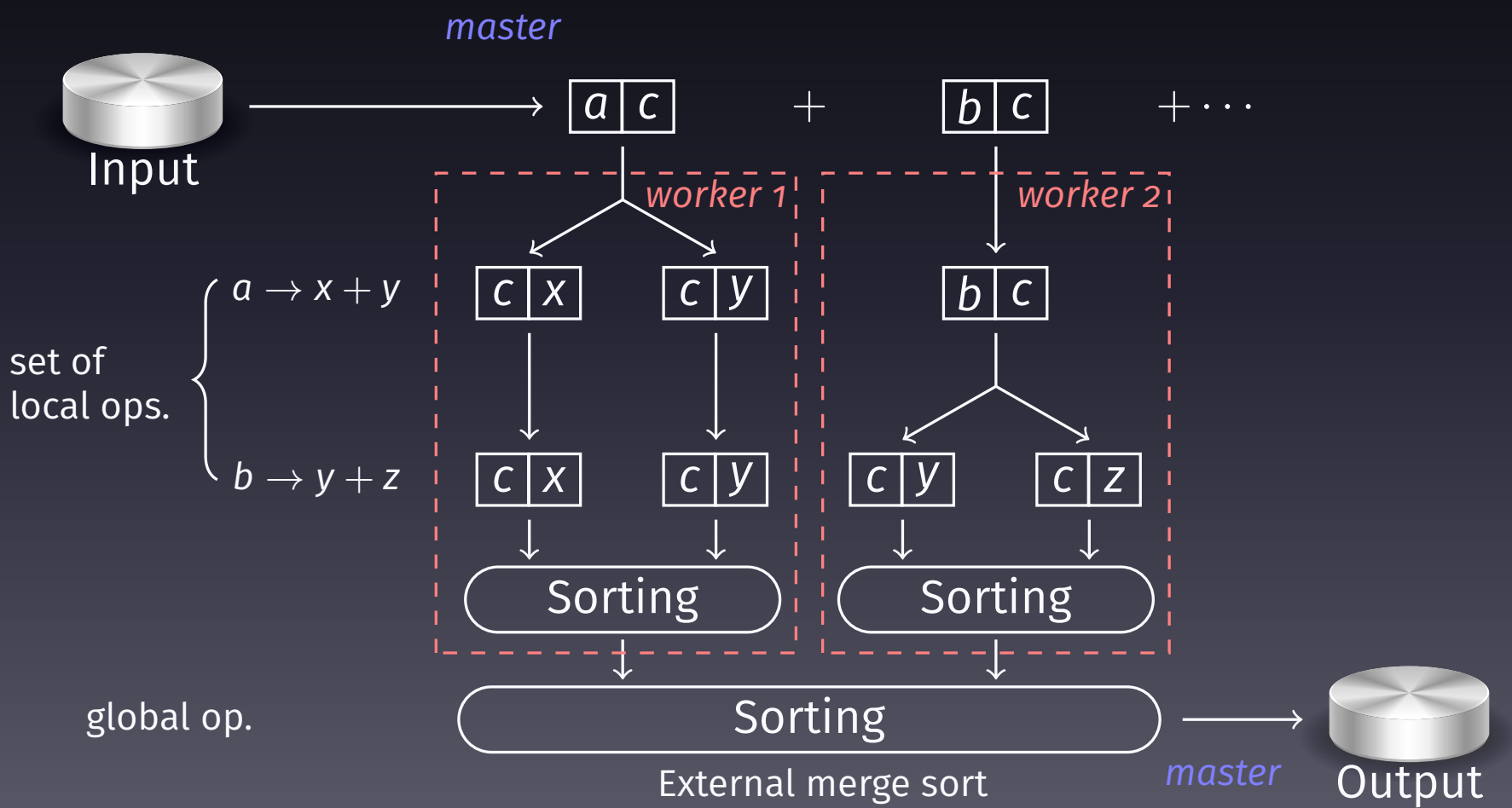
Local operations on each term
Global operations, e.g., sorting



Sequential access only. Works with disks ☺

Parallelization

Master-worker model



TFORM (POSIX Threads), ParFORM (MPI)

FORM as language

Imperative programming, term rewriting
Just write operations for every term

```
id x^n? = n * x^(n-1);
```

Replace $x^n \to nx^{n-1}$ *for* $n \in \mathbb{Z}$, *i.e.,* $\frac{\partial}{\partial x}$

FORM as language

Imperative programming, term rewriting
Just write operations for every term

```
id x^n? = n * x^(n-1);
```

Replace $x^n \to nx^{n-1}$ *for* $n \in \mathbb{Z}$, *i.e.,* $\frac{\partial}{\partial x}$

Powerful 'preprocessor'

```
#procedure derivative(x,m)
  #do i=1,`m'
    id `x'^n? = n * `x'^(n-1);
  #enddo
#endprocedure

#call derivative(y,2)
```

Define $\frac{\partial^m}{\partial x^m}$
x and m are parametrized

Perform $\frac{\partial^2}{\partial y^2}$

Code example (1)

example.frm

```
1 On highfirst;  * term order
2 Symbol a,b;
3 Local F = (a+b)^2;
4 Print;
5 .end
```

Code example (1)

example.frm

```
1 On highfirst;  * term order
2 Symbol a,b;
3 Local F = (a+b)^2;
4 Print;
5 .end
```

Run FORM as `form example.frm`

```
FORM 4.2.0 (Jul  6 2017, v4.2.0) 64-bits        Run: Thu Aug 17 18:10:24 2017
    On highfirst;  * term order
    Symbol a,b;
    Local F = (a+b)^2;
    Print;
    .end

Time =       0.00 sec    Generated terms =          3
               F         Terms in output =          3
                         Bytes used      =        108

   F =
      a^2 + 2*a*b + b^2;

  0.00 sec out of 0.00 sec
```

Code example (2)

```
1  On highfirst;   * term order
2  Symbol a,b,x;
3  Local F = a*x + x^2;
4
5  id x = a + b;
6
7  Print;
8  .sort
9
10 if (count(b,1) == 1);   * only for b^1
11   multiply a/b;
12 endif;
13
14 Print;
15 .end
```

Code example (2)

```
1  On highfirst;   * term order
2  Symbol a,b,x;
3  Local F = a*x + x^2;
4
5  id x = a + b;
6
7  Print;
8  .sort
9
10 if (count(b,1) == 1);   * only for b^1
11   multiply a/b;
12 endif;
13
14 Print;
15 .end
```

```
a^2 + a*b + a^2 + 2*a*b + b^2
```

```
F =
    2*a^2 + 3*a*b + b^2;
```

① Compile & Run

User controls sorting

Code example (2)

```
1  On highfirst;   * term order
2  Symbol a,b,x;
3  Local F = a*x + x^2;
4
5  id x = a + b;
6
7  Print;
8  .sort
9
10 if (count(b,1) == 1);   * only for b^1
11   multiply a/b;
12 endif;
13
14 Print;
15 .end
```

```
a^2 + a*b + a^2 + 2*a*b + b^2

F =
    2*a^2 + 3*a*b + b^2;

2*a^2 + 3*a^2 + b^2

F =
    5*a^2 + b^2;
```

① Compile & Run

User controls sorting

② Compile & Run

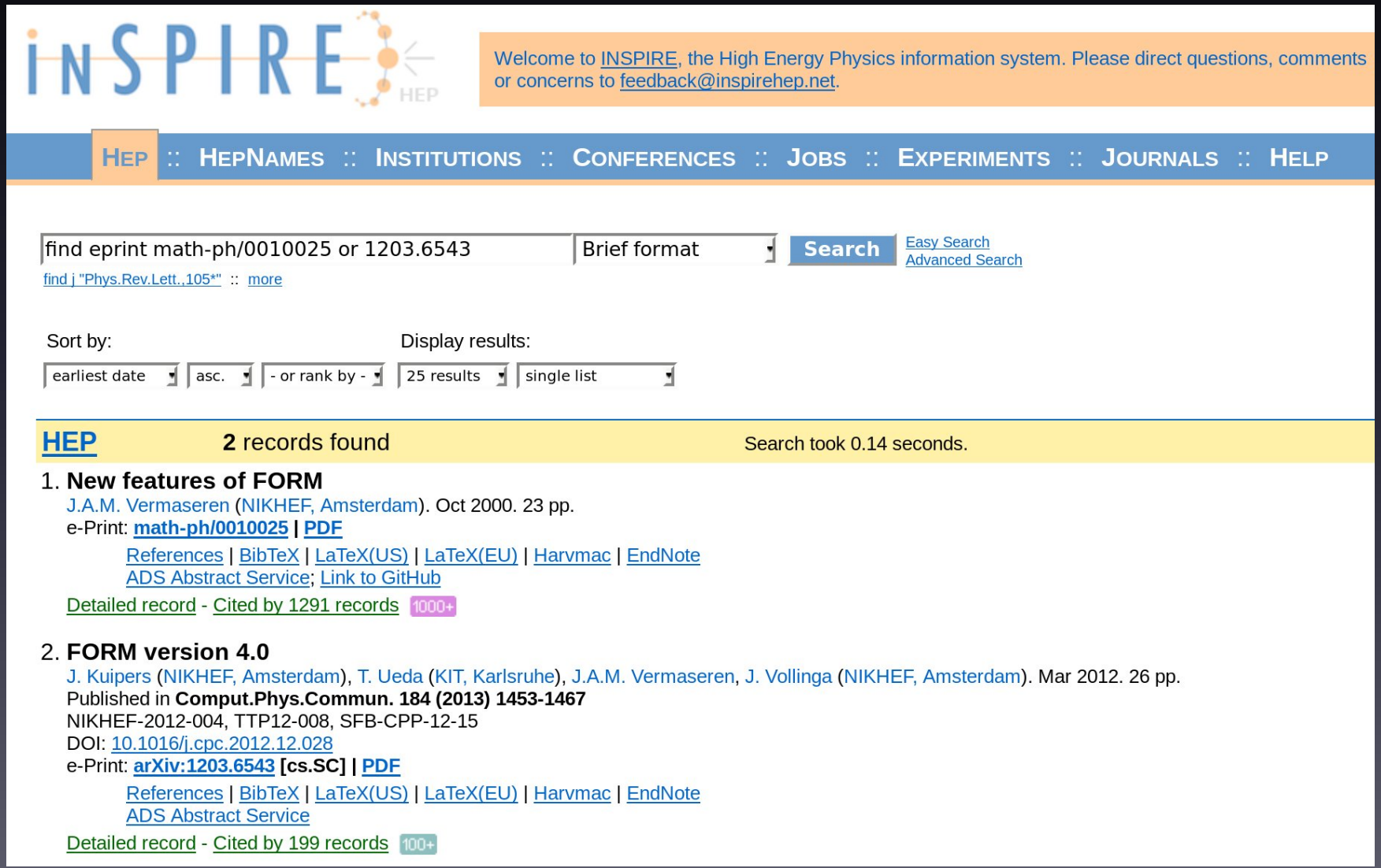
Preprocessor talks with processor

Compile-time optimization by the result of previous blocks

```
#define HaveX "0"

* Check whether expressions have x.
if (count(x,1));
  redefine HaveX "1";
endif;

.sort

#if `HaveX'
*  This part is skipped at compile-time
*  when there is no x in the expressions.
#endif
```

Physics applications

FORM has been used in many big computations



INSPIRE HEP

Welcome to INSPIRE, the High Energy Physics information system. Please direct questions, comments or concerns to feedback@inspirehep.net.

HEP :: HEPNAMES :: INSTITUTIONS :: CONFERENCES :: JOBS :: EXPERIMENTS :: JOURNALS :: HELP

find eprint math-ph/0010025 or 1203.6543 | Brief format | Search | Easy Search | Advanced Search

find j "Phys.Rev.Lett.,105*" :: more

Sort by: earliest date | asc. | - or rank by - | Display results: 25 results | single list

HEP **2** records found Search took 0.14 seconds.

1. **New features of FORM**
   J.A.M. Vermaseren (NIKHEF, Amsterdam). Oct 2000. 23 pp.
   e-Print: **math-ph/0010025** | **PDF**
   References | BibTeX | LaTeX(US) | LaTeX(EU) | Harvmac | EndNote
   ADS Abstract Service; Link to GitHub
   Detailed record - Cited by 1291 records 1000+

2. **FORM version 4.0**
   J. Kuipers (NIKHEF, Amsterdam), T. Ueda (KIT, Karlsruhe), J.A.M. Vermaseren, J. Vollinga (NIKHEF, Amsterdam). Mar 2012. 26 pp.
   Published in **Comput.Phys.Commun. 184 (2013) 1453-1467**
   NIKHEF-2012-004, TTP12-008, SFB-CPP-12-15
   DOI: 10.1016/j.cpc.2012.12.028
   e-Print: **arXiv:1203.6543 [cs.SC]** | **PDF**
   References | BibTeX | LaTeX(US) | LaTeX(EU) | Harvmac | EndNote
   ADS Abstract Service
   Detailed record - Cited by 199 records 100+

Physics applications

Because any choice of physics applications from 1291 citations would be biased, I just mention the keywords of our recent works:

FORCER and R*

- 👉 *TU's talk in ACAT 2016*
- 👉 *see Ruijl's talk*

They were the driving force of the developments of FORM 4.2

Outline

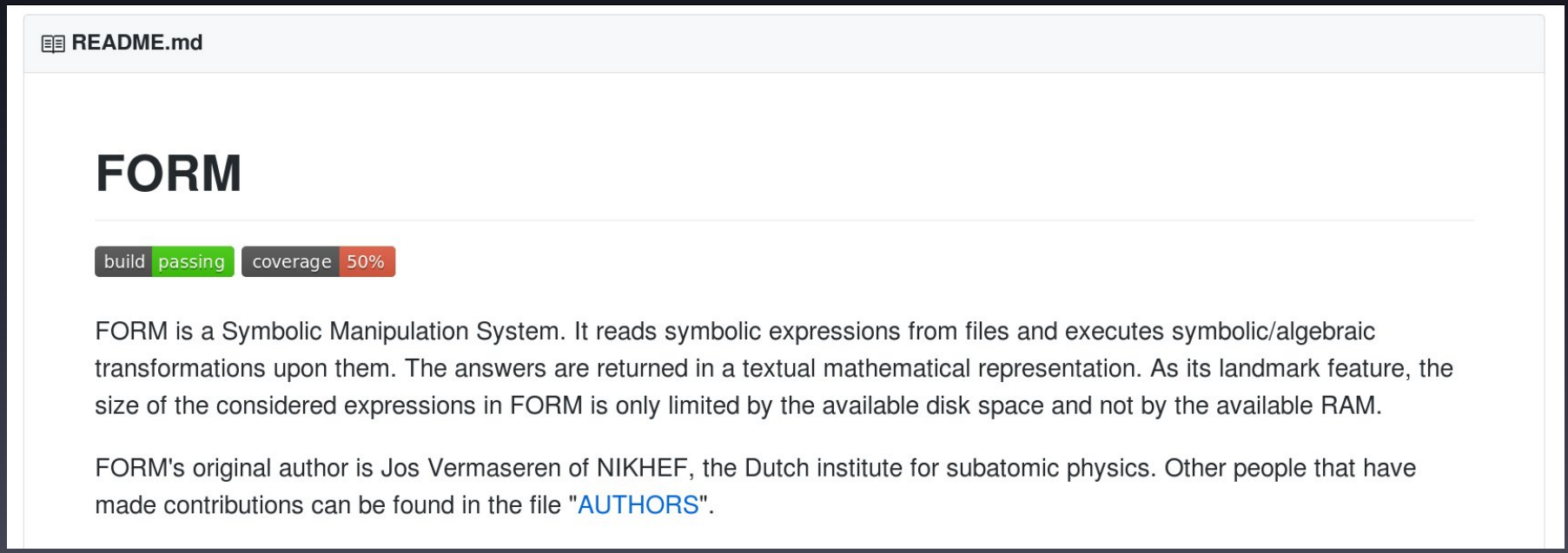
Form 4.2

v4.0	2012-03-30	Kuipers, TU, Vermaseren, Vollinga Polyomial factorization, rational polynomials, etc.
v4.1	2013-10-25	Kuipers, TU, Vermaseren Code output optimization, etc.
v4.2	2017-07-06	Ruijl, TU, Vermaseren Generating all matches, automatic expansion of rational polynomials, dictionaries, spectators, etc.

v4.2 contains more than 20 new features & 50 bugfixes

FORM in GitHub ecosystem

Hosted on GitHub since 2013-09-11
(Just before v4.1)


README.md

FORM

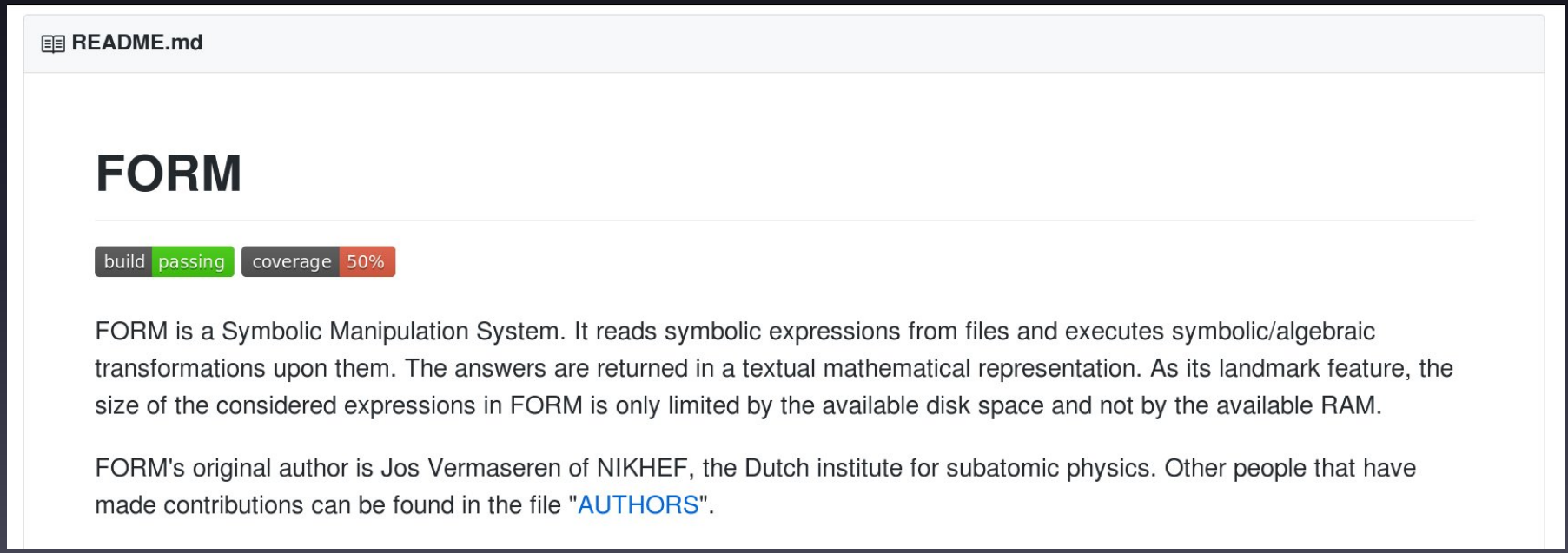
build passing coverage 50%

FORM is a Symbolic Manipulation System. It reads symbolic expressions from files and executes symbolic/algebraic transformations upon them. The answers are returned in a textual mathematical representation. As its landmark feature, the size of the considered expressions in FORM is only limited by the available disk space and not by the available RAM.

FORM's original author is Jos Vermaseren of NIKHEF, the Dutch institute for subatomic physics. Other people that have made contributions can be found in the file "AUTHORS".

FORM in GitHub ecosystem

Hosted on GitHub since 2013-09-11
(Just before v4.1)


README.md

FORM

build passing coverage 50%

FORM is a Symbolic Manipulation System. It reads symbolic expressions from files and executes symbolic/algebraic transformations upon them. The answers are returned in a textual mathematical representation. As its landmark feature, the size of the considered expressions in FORM is only limited by the available disk space and not by the available RAM.

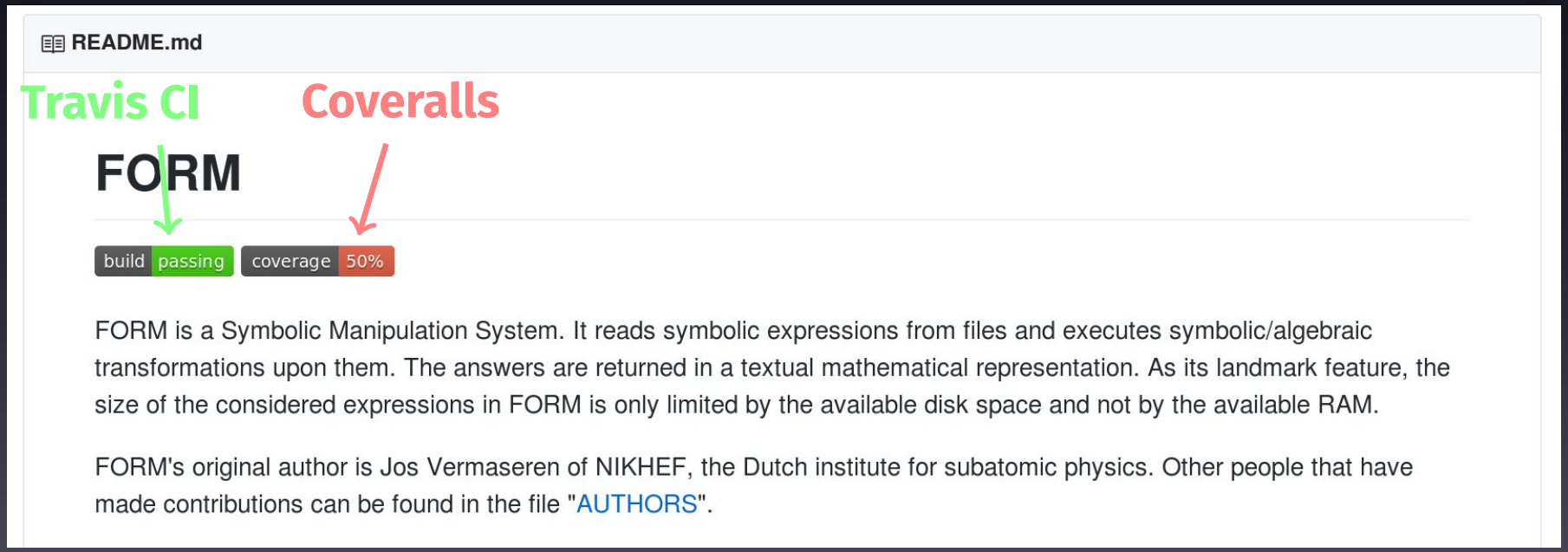
FORM's original author is Jos Vermaseren of NIKHEF, the Dutch institute for subatomic physics. Other people that have made contributions can be found in the file "AUTHORS".

Issue tracker (reporting bugs, feature requests, questions)

Wiki (release notes, installation notes)

FORM in GitHub ecosystem

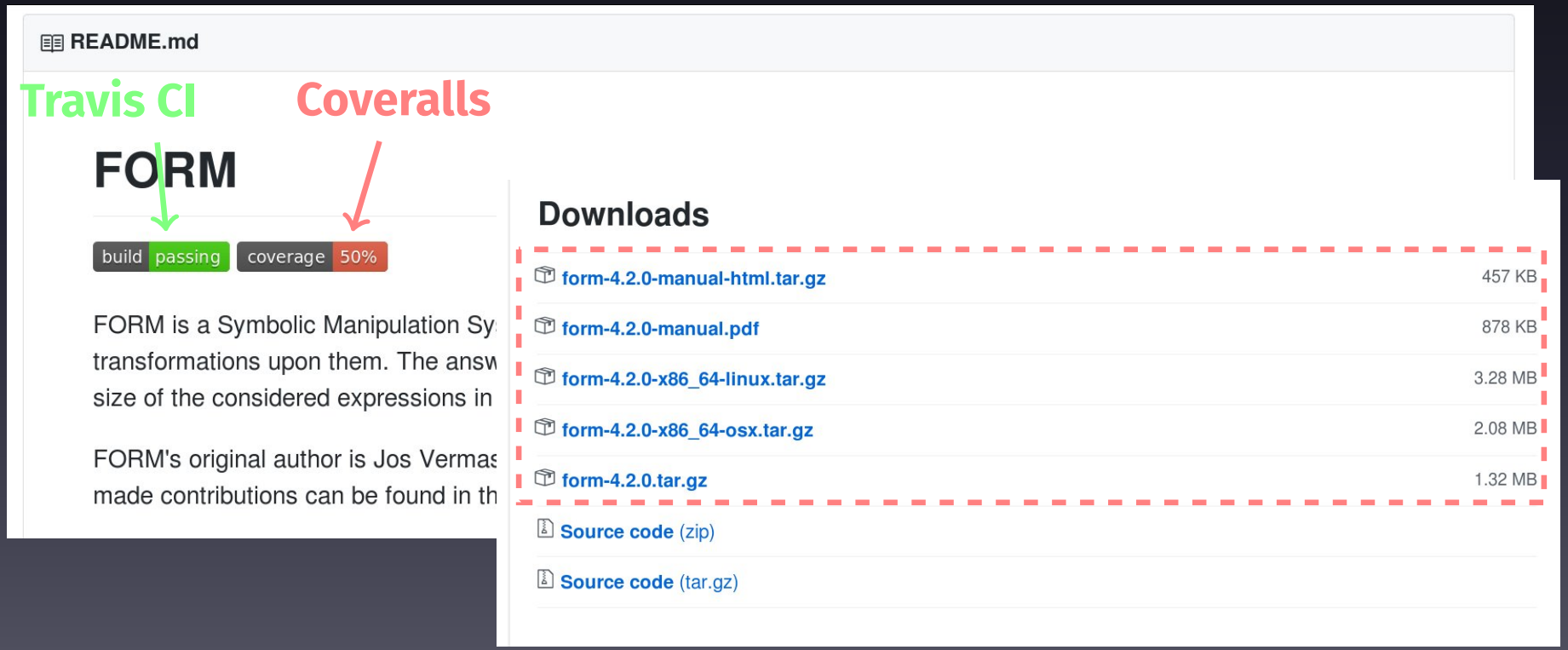
Hosted on GitHub since 2013-09-11
(Just before v4.1)



Every commit triggers
- Running test programs on Travis CI
- Code coverage test on Coveralls

FORM in GitHub ecosystem

Hosted on GitHub since 2013-09-11
(Just before v4.1)



Deployment from Travis CI to Github Release

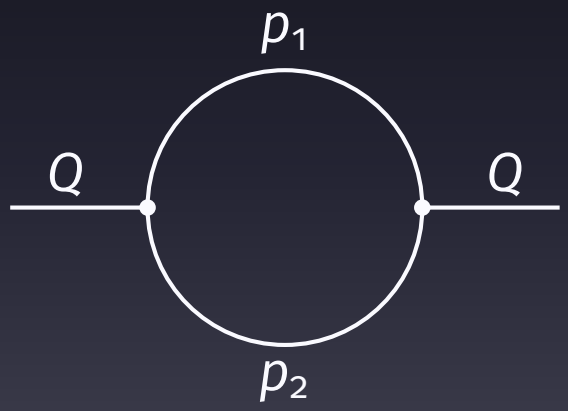
Graph manipulation

FORCER and R* program need graph manipulation

Graph manipulation

FORCER and R* program need graph manipulation

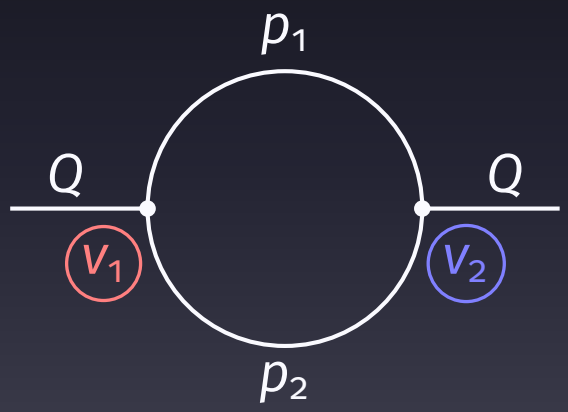
Undirected graph:



Graph manipulation

FORCER and R* program need graph manipulation

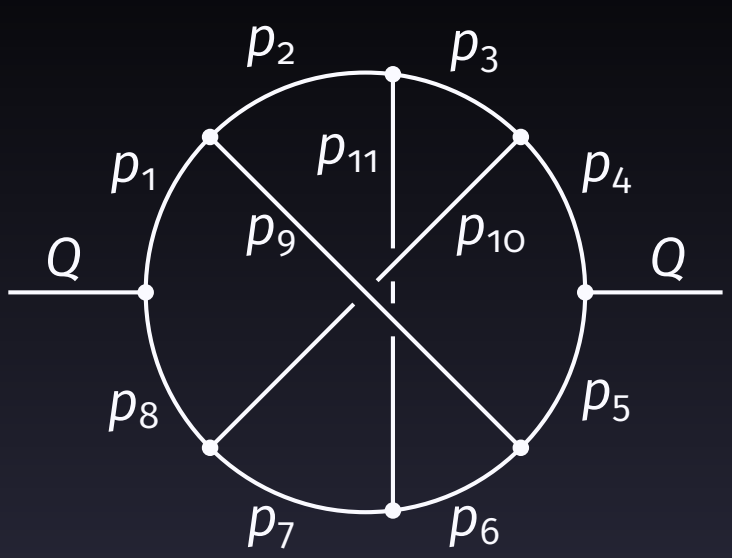
Undirected graph:

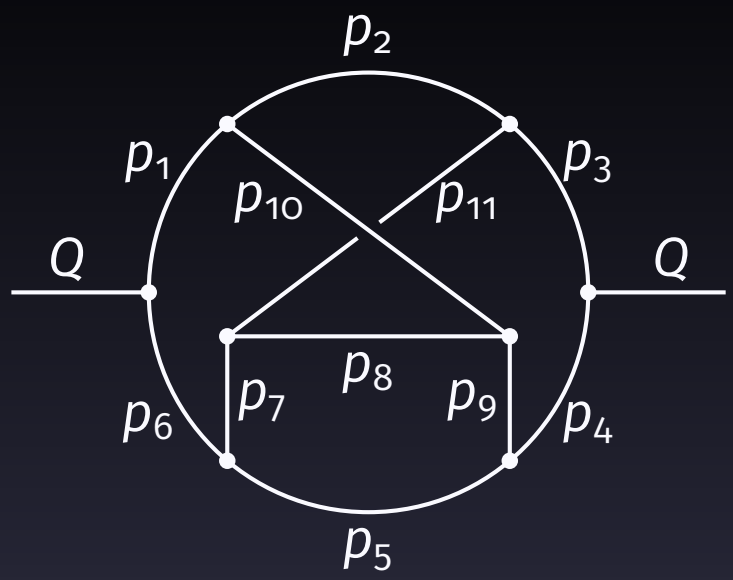



```
CFunction vx(symmetric);   * commuting & symmetric
Vector Q,p1,...,p99;

Local F = vx(Q,p1,p2) * vx(Q,p1,p2);
```

v_1 v_2

Graph isomorphism





Q. Are they equivalent?

```
Local F1 = vx(Q,p1,p8)*vx(p1,p2,p9)*vx(p2,p3,p11)*
           vx(p3,p4,p10)*vx(Q,p4,p5)*vx(p5,p6,p9)*
           vx(p6,p7,p11)*vx(p7,p8,p10);

Local F2 = vx(Q,p1,p6)*vx(p1,p2,p10)*vx(p2,p3,p11)*
           vx(Q,p3,p4)*vx(p4,p5,p9)*vx(p5,p6,p7)*
           vx(p7,p8,p11)*vx(p8,p9,p10);
```

Graph isomorphism

```
1  CFunction map;  * for isomorphic mapping
2  CFunction vx(symmetric);
3  Vector Q,p1,...,p11,q1,...,q11;
4
5  Local F1 = vx(Q,p1,p8)*vx(p1,p2,p9)*vx(p2,p3,p11)*
6             vx(p3,p4,p10)*vx(Q,p4,p5)*vx(p5,p6,p9)*
7             vx(p6,p7,p11)*vx(p7,p8,p10);
8
9  Local F2 = vx(Q,p1,p6)*vx(p1,p2,p10)*vx(p2,p3,p11)*
10            vx(Q,p3,p4)*vx(p4,p5,p9)*vx(p5,p6,p7)*
11            vx(p7,p8,p11)*vx(p8,p9,p10);
12
13 id vx(Q,q1?,q8?)*vx(q1?,q2?,q9?)*vx(q2?,q3?,q11?)*
14    vx(q3?,q4?,q10?)*vx(Q,q4?,q5?)*vx(q5?,q6?,q9?)*
15    vx(q6?,q7?,q11?)*vx(q7?,q8?,q10?)
16    = map(q1,...,q11);
17 Print;
18 .end
```

Pattern with q1?,...,q11?

Graph isomorphism

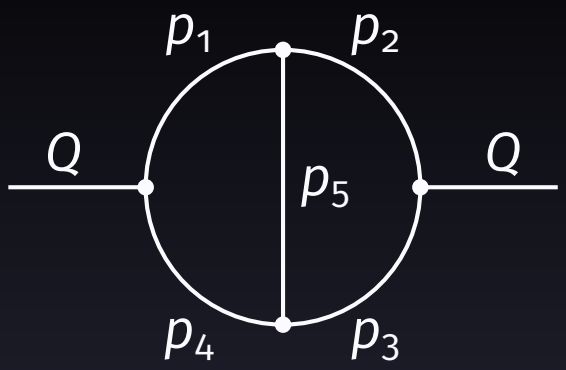
```
1  CFunction map;  * for isomorphic mapping
2  CFunction vx(symmetric);
3  Vector Q,p1,...,p11,q1,...,q11;
4
5  Local F1 = vx(Q,p1,p8)*vx(p1,p2,p9)*vx(p2,p3,p11)*
6             vx(p3,p4,p10)*vx(Q,p4,p5)*vx(p5,p6,p9)*
7             vx(p6,p7,p11)*vx(p7,p8,p10);
8
9  Local F2 = vx(Q,p1,p6)*vx(p1,p2,p10)*vx(p2,p3,p11)*
10            vx(Q,p3,p4)*vx(p4,p5,p9)*vx(p5,p6,p7)*
11            vx(p7,p8,p11)*vx(p8,p9,p10);
12
13 id vx(Q,q1?,q8?)*vx(q1?,q2?,q9?)*vx(q2?,q3?,q11?)*
14    vx(q3?,q4?,q10?)*vx(Q,q4?,q5?)*vx(q5?,q6?,q9?)*
15    vx(q6?,q7?,q11?)*vx(q7?,q8?,q10?)
16    = map(q1,...,q11);
17 Print;
18 .end
```

Pattern with q1?,...,q11?

```
   F2 =
      map(p6,p7,p11,p3,p4,p9,p10,p1,p5,p2,p8);
```

Yes. Got the isomorphic mapping

Graph automorphism

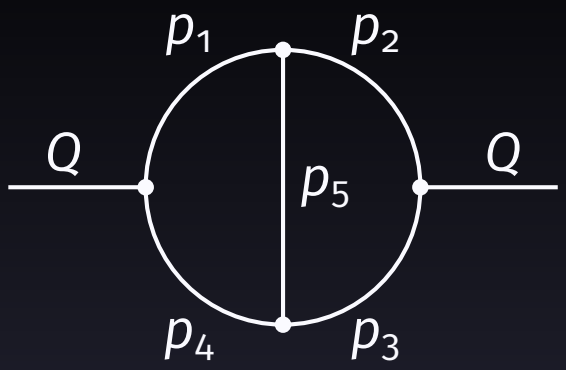


`id all` generates all matches

(v4.2)

```
1  CFunction map;
2  CFunction vx(symmetric);
3  Vector Q,p1,...,p5,q1,...,q5;
4
5  Local F = vx(Q,p1,p4)*vx(p1,p2,p5)*vx(Q,p2,p3)*
6            vx(p3,p4,p5);
7
8  id all,vx(Q,q1?,q4?)*vx(q1?,q2?,q5?)*vx(Q,q2?,q3?)*
9         vx(q3?,q4?,q5?)
10        = map(q1,...,q5);
11 Print +s;
12 .end
```

Graph automorphism



`id all` generates all matches

(v4.2)

```
F =
    + map(p1,p2,p3,p4,p5)
    + map(p2,p1,p4,p3,p5)
    + map(p3,p4,p1,p2,p5)
    + map(p4,p3,p2,p1,p5)
   ;
```

Exhibits $Z_2 \times Z_2$ symmetry

Creative (ab)use of features

```
id vx(Q,q1?,q8?)*vx(q1?,q2?,q9?)*vx(q2?,q3?,q11?)*
   vx(q3?,q4?,q10?)*vx(Q,q4?,q5?)*vx(q5?,q6?,q9?)*
   vx(q6?,q7?,q11?)*vx(q7?,q8?,q10?)
   = map(q1,...,q11);
```

```
id all,vx(Q,q1?,q4?)*vx(q1?,q2?,q5?)*vx(Q,q2?,q3?)*
       vx(q3?,q4?,q5?)
       = map(q1,...,q5);
```

The patterns are rather static. Better way?

Creative (ab)use of features

“Dictionary” at compile-time (v4.2)

```
#$graph1 = F1;
#opendictionary wild
  #do i=1,11
    #add p`i': "q`i'?"
  #enddo
#closedictionary
#usedictionary wild($)
  id `$graph1' = map(q1,...,q11);
#closedictionary
```

Dictionary: e.g.,

mu1 $\rightarrow$ “\mu_1”

Translates

```
Local F1 = vx(Q,p1,p8)*vx(p1,p2,p9)*vx(p2,p3,p11)*
           vx(p3,p4,p10)*vx(Q,p4,p5)*vx(p5,p6,p9)*
           vx(p6,p7,p11)*vx(p7,p8,p10);
```

into

```
id vx(Q,q1?,q8?)*vx(q1?,q2?,q9?)*vx(q2?,q3?,q11?)*
   vx(q3?,q4?,q10?)*vx(Q,q4?,q5?)*vx(q5?,q6?,q9?)*
   vx(q6?,q7?,q11?)*vx(q7?,q8?,q10?)
   = map(q1,...,q11);
```

Creative (ab)use of features

Replacement on the LHS at run-time (v4.2)

```
$graph1 = F1;
id $graph1 * replace_(<p1,q1?>,...,<p11,q11?>)
   = map(q1,...,q11);
```

Translates

```
Local F1 = vx(Q,p1,p8)*vx(p1,p2,p9)*vx(p2,p3,p11)*
           vx(p3,p4,p10)*vx(Q,p4,p5)*vx(p5,p6,p9)*
           vx(p6,p7,p11)*vx(p7,p8,p10);
```

into

```
id vx(Q,q1?,q8?)*vx(q1?,q2?,q9?)*vx(q2?,q3?,q11?)*
   vx(q3?,q4?,q10?)*vx(Q,q4?,q5?)*vx(q5?,q6?,q9?)*
   vx(q6?,q7?,q11?)*vx(q7?,q8?,q10?)
   = map(q1,...,q11);
```

Conclusion

Conclusion

FORM is designed for symbolic manipulation of very big expressions, required by perturbative QFT

Conclusion

FORM is designed for symbolic manipulation of very big expressions, required by perturbative QFT

Used in many cutting edge computations

Conclusion

FORM is designed for symbolic manipulation of very big expressions, required by perturbative QFT

Used in many cutting edge computations

New features in FORM 4.2 help users to make ever smarter programs

Conclusion

FORM is designed for symbolic manipulation of very big expressions, required by perturbative QFT

Used in many cutting edge computations

New features in FORM 4.2 help users to make ever smarter programs

Questions? Or  ?