Automation of Feynman-diagrammatic Calculations at the One-Loop Level

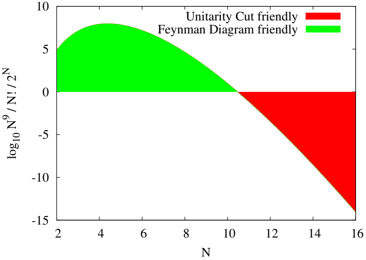
in collaboration with
T. Binoth, A. Guffanti, J.-Ph. Guillet, G. Heinrich, S. Karg, N. Kauer & J. Reuter

Mini-workshop on fixed order multi-leg automatic NLO calculations, Wuppertal 2009

Outline

Motivation

- Many processes at NLO precision required for the LHC
- Both Feynman Diag. and Unitarity Cuts look promising for $N \approx 6$
- Complex calculations $\Rightarrow$ Automation necessary
- Our approach:
  - Feynman Diagram based
  - Tensor Integral Reduction
  - Semi-Numerical



[Binoth]

A One-Loop Amplitude's Anatomy

Colour	**Integral**	**Lorentz Objects**
t^A_{ij}, f^{ABC}	$\int d^n k \frac{k^\mu k^\nu \cdots}{((k+r1)^2-m_1^2)\cdots}$	$\langle p_1 \vert \gamma^\mu \not p \cdots \vert p_2 \rangle, p^\mu, \ldots$
Tensor Reduction	Tensor Reduction	
Basis Tensors $\vert c_1\rangle, \vert c_2\rangle, \ldots$	Form Factors A^r_N, B^r_N, C^r_N	Spinor Products $\langle pq\rangle, [pq]$

have numerical representation

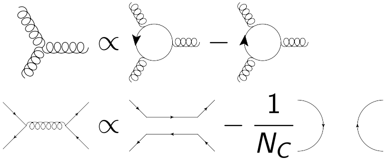
Colour Structure

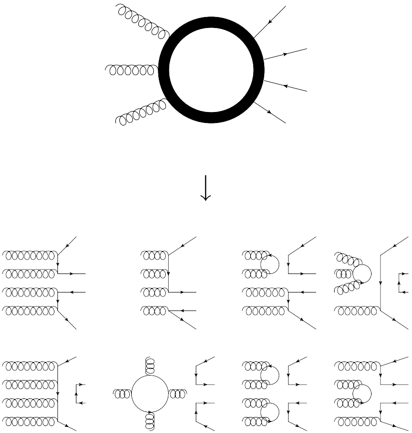
Colour basis is created by this

Algorithm

Generate all (possibly disconnected) bubble-free diagrams from only $q\bar{q}g$ vertices and $q\bar{q}$ propagators

SU(N_C) Diagrams reduced by





Colour Structure (II)

- Split basis tensors $|c_1\rangle, |c_2\rangle, \ldots$ off the amplitude
- Calculate matrices $\langle c_i|c_j\rangle$ and $\langle c_i|T_p \cdot T_q|c_j\rangle / T_p^2$ once
- Worst $2 \to 4$ case (six gluons): 265 basis elements but typically basis much smaller
- No colour ordering, just decomposition

$\Rightarrow$ Colour is not the bottleneck.

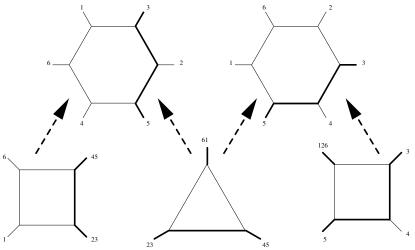
Colour Structure (II)

- Split basis tensors $|c_1\rangle, |c_2\rangle, \ldots$ off the amplitude
- Calculate matrices $\langle c_i|c_j\rangle$ and $\langle c_i|T_p \cdot T_q|c_j\rangle / T_p^2$ once
- Worst $2 \to 4$ case (six gluons): 265 basis elements but typically basis much smaller
- No colour ordering, just decomposition

$\Rightarrow$ Colour is not the bottleneck.

Integral Reduction

- Integral reduction done by GOLEM95 library [arXiv:0810.0992]
- Avoids numerical problems due to inverse Gram determinants
  - Reduction to scalar integrals in the bulk
  - Direct numerical integration at problematic points (using efficient repr. of Feynman parameter integrals)
- Allows for reuse of form factors in pinched topologies



Typically most of the run time is spent in evaluation of integrals.

Coefficients

- Coefficients expressed in terms of spinor products
- Can become quite large, e.g. $\mathcal{O}(MB)$ for hexagon
- Constitute significant part of the run time

Problem: Compiler not designed for large expressions

- Often simply too big for compiler
- Overlooks optimizations (commutativity, distributivity, ...)

Solution: Preprocessor `haggies` turns expressions into optimized programs

Coefficients

- Coefficients expressed in terms of spinor products
- Can become quite large, e.g. $\mathcal{O}(MB)$ for hexagon
- Constitute significant part of the run time

Problem: Compiler not designed for large expressions

- Often simply too big for compiler
- Overlooks optimizations (commutativity, distributivity, ...)

Solution: Preprocessor `haggies` turns expressions into optimized programs

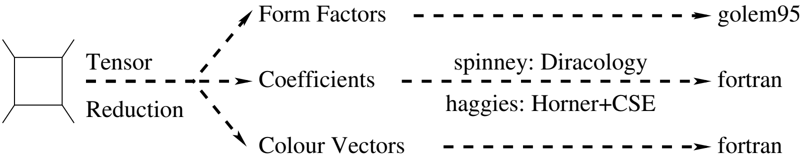
Coefficients

- Coefficients expressed in terms of spinor products
- Can become quite large, e.g. $\mathcal{O}(MB)$ for hexagon
- Constitute significant part of the run time

Problem: Compiler not designed for large expressions

- Often simply too big for compiler
- Overlooks optimizations (commutativity, distributivity, ...)

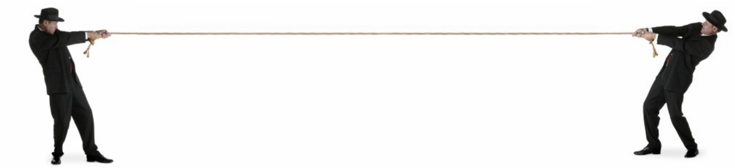
Solution: Preprocessor `haggies` turns expressions into optimized programs



Automation

Any Task — No Automation

Full Automation — One Task



We better stay somewhere inbetween!

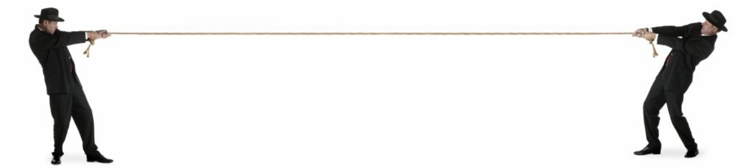
Generation of one-loop matrix element + subtraction terms from

- particle configuration (initial/final state)
- particle content $\subseteq$ Standard Model (to start with)
- optional set of approximations (e.g. $m_i = 0$, $V_{ff'} = \delta_{ff'}$, ...)

Automation

Any Task — No Automation

Full Automation — One Task

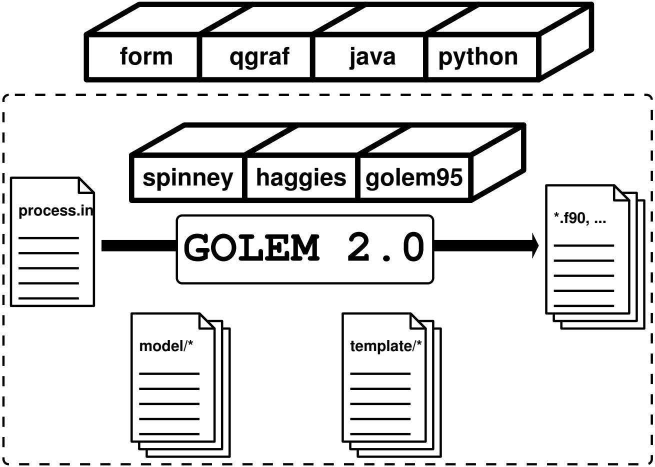


We better stay somewhere inbetween!

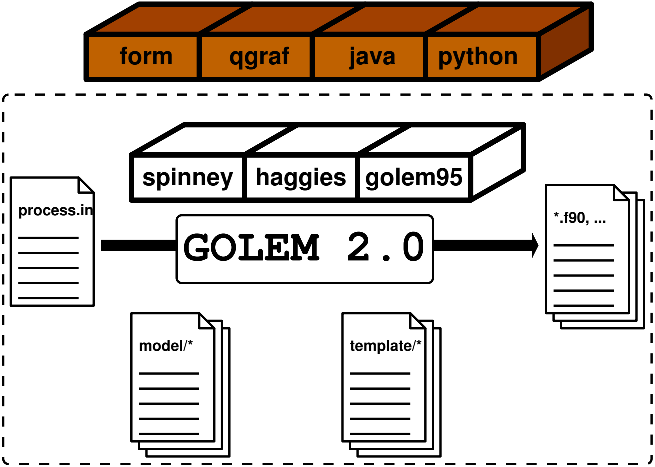
Generation of one-loop matrix element + subtraction terms from

- particle configuration (initial/final state)
- particle content $\subseteq$ Standard Model (to start with)
- optional set of approximations (e.g. $m_i = 0$, $V_{ff'} = \delta_{ff'}$, ...)

GOLEM: An Overview

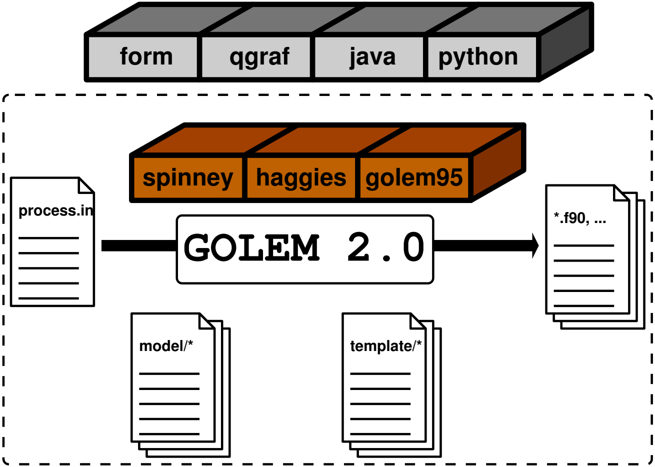


GOLEM: An Overview



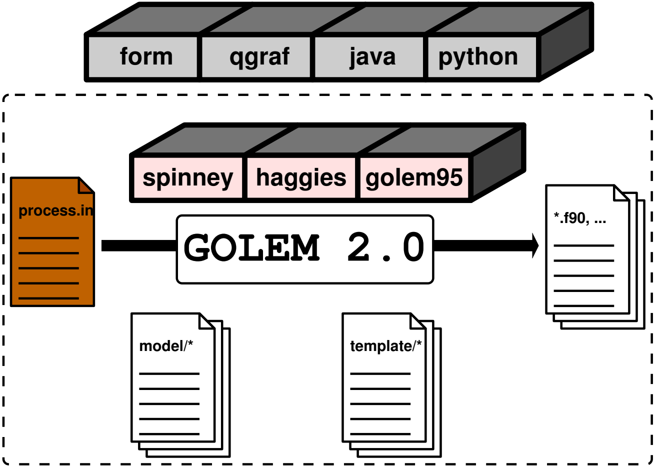
System requirements: free standard software only

GOLEM: An Overview



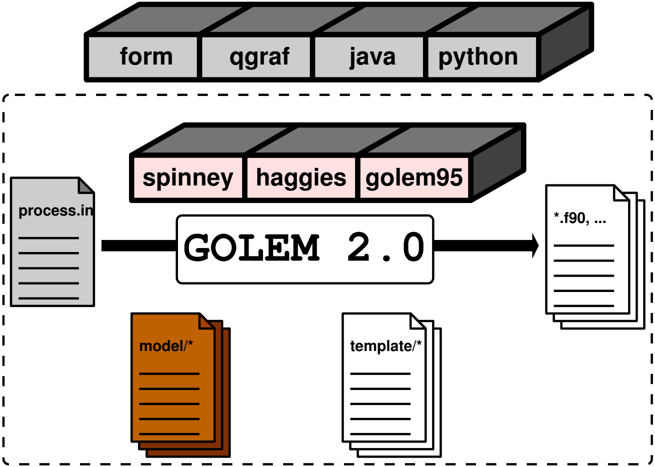
Independent modules $\Rightarrow$ can also be used stand-alone

GOLEM: An Overview



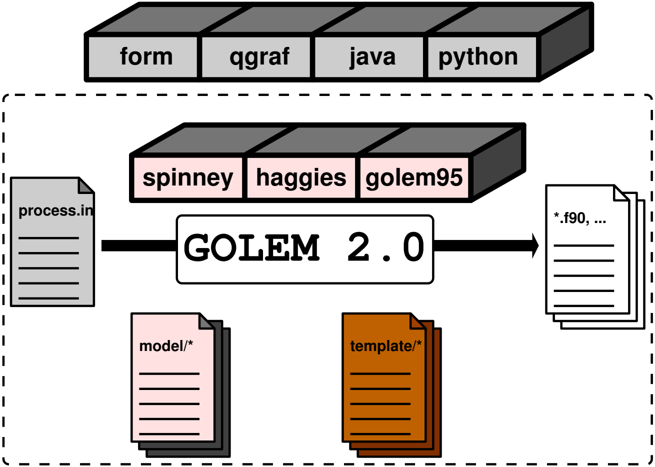
Single input file (key-value list) specifies process

GOLEM: An Overview



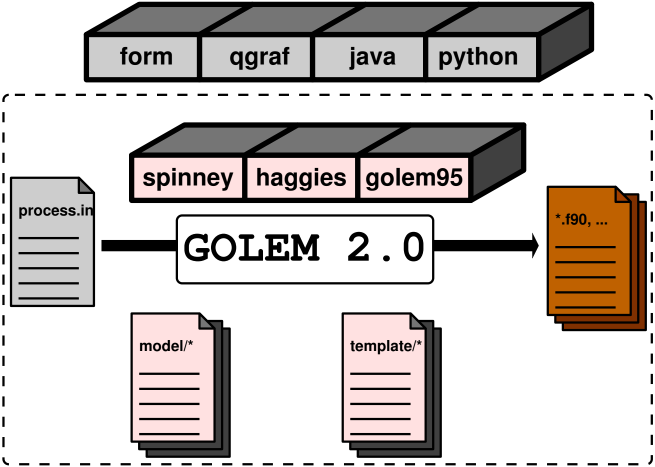
Model files (SM provided) easily extensible

GOLEM: An Overview



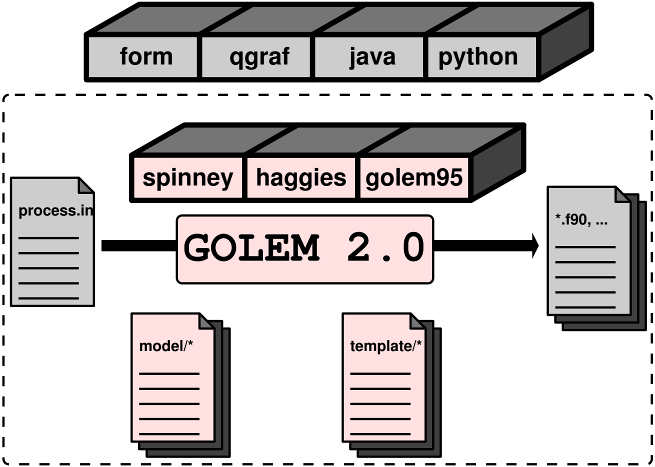
Templates to define the output (currently F90)

GOLEM: An Overview



Output: helicity amplitudes (born part, virtual part, int. dipoles)

GOLEM: An Overview



spinney

squared amplitude	helicity amplitude
spinor traces	spinor chains
$\longrightarrow$ Mandelstam variables	$\longrightarrow$ spinor products
FORM	???

- Packages available but not for FORM: S@M [Maitre, Mastrolia 07]
- spinney: FORM library, similar functionality as S@M
- in GOLEM used to reduce spinor chains
- can be used independently

squared amplitude	helicity amplitude
spinor traces	spinor chains
$\longrightarrow$ Mandelstam variables	$\longrightarrow$ spinor products
FORM	???

- Packages available but not for FORM: S@M [Maitre, Mastrolia 07]
- spinney: FORM library, similar functionality as S@M
- in GOLEM used to reduce spinor chains
- can be used independently

spinney

squared amplitude	helicity amplitude
spinor traces	spinor chains
$\rightarrow$ Mandelstam variables	$\rightarrow$ spinor products
FORM	spinney

- Packages available but not for FORM: S@M [Maitre, Mastrolia 07]
- spinney: FORM library, similar functionality as S@M
- in `GOLEM` used to reduce spinor chains
- can be used independently

spinney

```
vectors k1, k2, p3, p4, k3, k4;
indices mu, nu; symbol m;
#include- spinney.hh
local Amp = UbarSpb(k2) * Sm(mu) * USpa(k1) *
        d(mu, nu) *
        UbarSpb(p3, +1) * Sm(nu) * USpa(p4, -1);
#call LightConeDecomposition(p3, k3, k2, m)
#call LightConeDecomposition(p4, k4, k2, m)
#call tHooftAlgebra
#call SpCollect
#call SpContractMetrics
#call SpContract
#call SpOpen
id SpDenominator(m?) = 1/m;
print;
.end
```

spinney

Output

```
Amp =
   - 2*Spa2(k1,k4)*Spb2(k2,k3);
```

$$\mathrm{Amp} = -2\langle k_1 k_4\rangle[k_2 k_3]$$

- "in principle" suitable for numerical evaluation
- toy example $\rightarrow$ real world: expressions much larger
- expressions must be optimized
  - save arithmetic operations (efficiency)
  - reduce complexity for compiler

GOALS

- on the algebra side:
  - save function calls
  - save arithmetic operations
  - reduce complexity for compiler
- on the interface side:
  - large class of input expressions
  - target independent (syntax, type system, ...)

SOLUTION:

- multivariate Horner scheme [Ceberio, Kreinovich 03]
- common subexpression elimination [Aho, Sethi, Ullman 86]
- economic variable allocation [Poletto et al. 97]
- built-in rule based type checker
- regex based syntax transformations

Examples: helicity amplitude: $u_{(-)}\bar{u}_{(+)} \to W^-_{(-)} W^+_{(+)} g_{(+)} g_{(+)}$

Sum of all LO diagrams

	multiplications	additions	form factor calls
unoptimized	26,558	6,979	0
optimized	3,660	1,032	0
savings	86%	85%	—

A single NLO diagram (5 point, rank 3)

	multiplications	additions	form factor calls
unoptimized	161,121	6,794	2,267
optimized	5,202	3,140	42
savings	97%	54%	98%

Summary

- Feynman Diagrams can provide efficient implementation of $2 \to 4$ at one-loop.
- `GOLEM` provides automation of (partonic) one-loop matrix element
- consists of independent modules:
  - `GOLEM95`: tensor integrals
  - spinney: spinor algebra
  - haggies: expression crunching
  - some Python "glue"
- `GOLEM95` already public
- spinney and haggies follow soon

Summary

- Feynman Diagrams can provide efficient implementation of $2 \to 4$ at one-loop.
- `GOLEM` provides automation of (partonic) one-loop matrix element
- consists of independent modules:
  - `GOLEM95`: tensor integrals
  - spinney: spinor algebra
  - haggies: expression crunching
  - some Python “glue”
- `GOLEM95` already public
- spinney and haggies follow soon

Summary

- Feynman Diagrams can provide efficient implementation of $2 \rightarrow 4$ at one-loop.
- GOLEM provides automation of (partonic) one-loop matrix element
- consists of independent modules:
  - GOLEM95: tensor integrals
  - spinney: spinor algebra
  - haggies: expression crunching
  - some Python "glue"
- GOLEM95 already public
- spinney and haggies follow soon

Summary

- Feynman Diagrams can provide efficient implementation of $2 \to 4$ at one-loop.
- `GOLEM` provides automation of (partonic) one-loop matrix element
- consists of independent modules:
  - `GOLEM95`: tensor integrals
  - spinney: spinor algebra
  - haggies: expression crunching
  - some Python "glue"
- `GOLEM95` already public
- spinney and haggies follow soon

Outlook

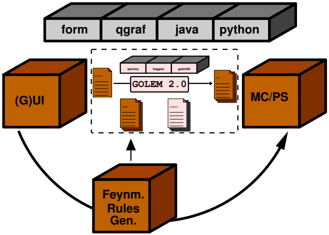
What's next? (time ordered)

- SM physics applications
  - $gg \to b\bar{b}b\bar{b}$
  - $pp \to VVjj$
- seamless integration
  - user interface
  - Monte Carlo and parton shower
  - Feynman rules
- BSM physics

Outlook

What's next? (time ordered)

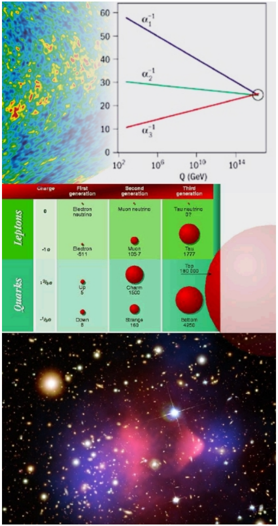
- SM physics applications
  - $gg \to b\bar{b}b\bar{b}$
  - $pp \to VVjj$
- seamless integration
  - user interface
  - Monte Carlo and parton shower
  - Feynman rules
- BSM physics

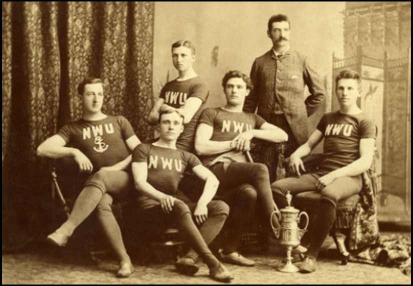


Outlook

What's next? (time ordered)

- SM physics applications
  - $gg \to b\bar{b}b\bar{b}$
  - $pp \to VVjj$
- seamless integration
  - user interface
  - Monte Carlo and parton shower
  - Feynman rules
- BSM physics





Thank You