



O'Mega: Optimal Monte-Carlo Event Generation Amplitudes

Thorsten Ohl
— TU Darmstadt —
<ohl@hep.tu-darmstadt.de>

Padova, May 2000



1	DAGs of Feynman Diagrams	2
	○ Perturbative Complexity ○ Prior Art	
	○ Redundancy ○ Topologies ○ Elimination	
2	O'Mega	10
	○ Mission ○ Features ○ Architecture ○ Models	
	○ Targets ○ Results ○ Shortcomings	
3	Outlook	19
	○ Short Term Progress ○ Long Term Progress	



1	DAGs of Feynman Diagrams	2
	Perturbative Complexity	
	Prior Art	
	Redundancy	
	Topologies	
	Elimination	
2	O'Mega	10
3	Outlook	19



There are

$$F(n) = (2n - 5)!! = (2n - 5) \cdot (2n - 7) \cdot \dots \cdot 3 \cdot 1$$

tree Feynman diagrams w/ n legs in vanilla ϕ^3 -theory:

n $2 \rightarrow (n - 2)$	$P(n)$ $2^{n-1} - n - 1$	$F(n)$ $(2n - 5)!!$
4	3	3
5	10	15
6	25	105
7	56	945
8	119	10395
9	246	135135
10	501	2027025
11	1012	34459425
12	2035	654729075

But there are only

$$P(n) = \frac{2^n - 2}{2} - n = 2^{n-1} - n - 1$$

independent internal momenta in these diagrams. This grows **much** slower than $F(n)$.



There are no closed formulae for more complicated (i. e. realistic) theories, but **empirical evidence** (from **O'Mega** & other programs) suggests

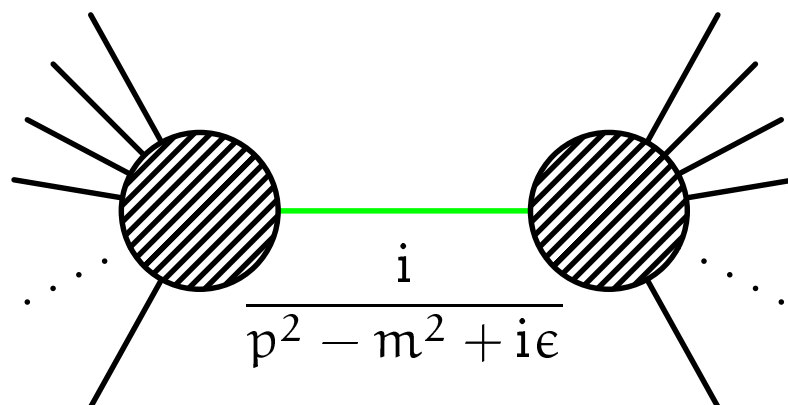
$$F^*(n) \propto (C \cdot n)!!$$

$$P^*(n) \propto 10^{n/2}$$

unless flavors are chosen w/ the objective to reduce the number of diagrams (e. g. CC10 vs. the 56 diagrams of $e^+ e^- \rightarrow e^+ \nu_e e^- \bar{\nu}_e$).

☹ realistic calculations (almost) **impossible** for $n \geq 9$, i. e. radiative corrections to **WW-(re)scattering** & other processes w/ **quartic gauge boson couplings**, when forced to calculate **$O(10^5)$** Feynman diagrams.

😊 I would not give this talk if there was no way to reduce **$F^*(n)$** to **$P^*(n)$** : Instead of calculating Feynman diagrams individually, construct the amplitude from the **$P^*(n)$ off-shell wavefunctions**:





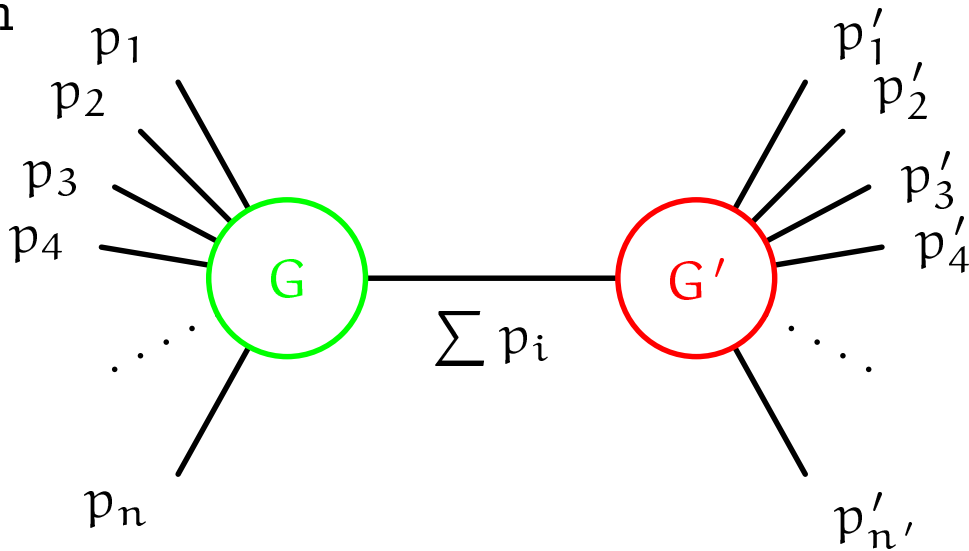
- ALPHA (Caravaglios & Moretti):
 - observation: tree level scattering amplitude is Legendre transform of Lagrangian
- 😊 can be performed numerically, since there are only $P^*(n)$ independent variables
- HELAC (Kanaki & Papadopoulos):
 - ALPHA algorithm can be reformulated as recursive numerical solution of Schwinger-Dyson equations
 - marginally less efficient, b/c it breaks more permutation symmetries than ALPHA

Shortcomings:

- strictly numerical
- 😞 perturbative renormalization machinery unavailable
- 😊 Lattice gauge theory methods remain an option (Moretti) ...
- hardcoded Lagrangians
- 😞 difficult to extent by users
- 😊 difficult to extent by users
- 😊 can easily be changed by the authors



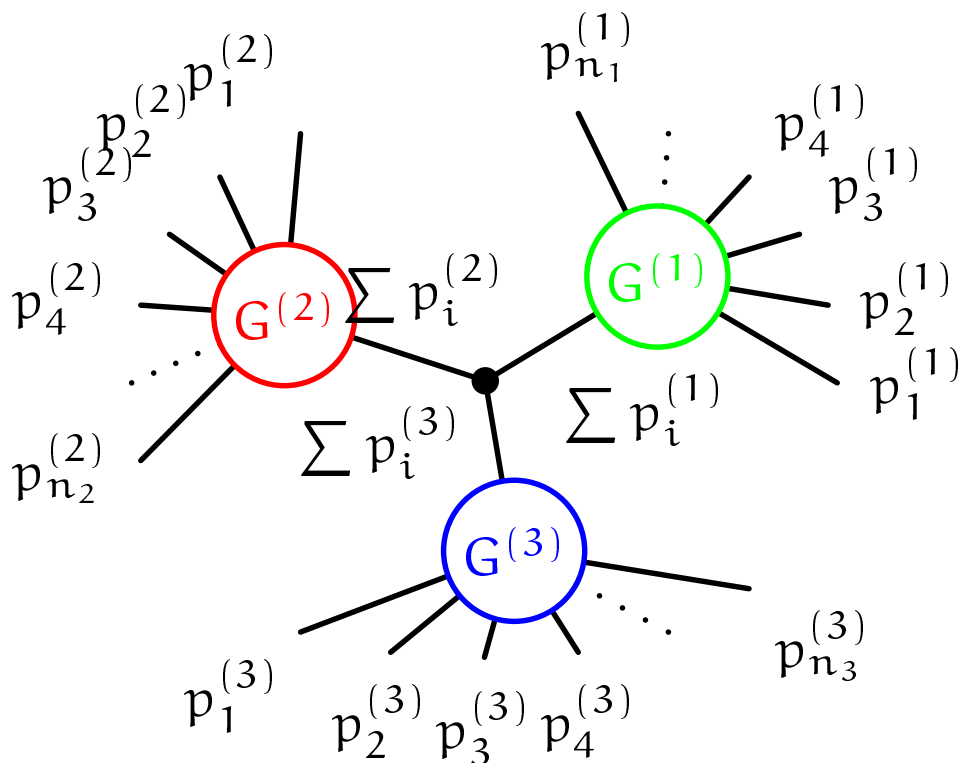
Indeed, in



each of the $(2n - 3)!!$ diagrams in G is multiplied by the $(2n' - 3)!!$ diagrams in G' .

☹️ A naive construction of the amplitude by joining subdiagrams along propagators leads to **double counting** of Feynman diagrams.

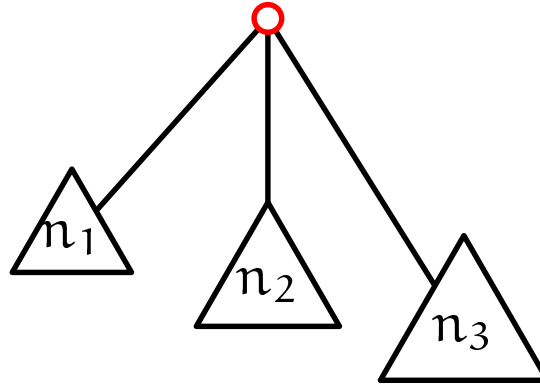
😊 Join at vertices instead:





Non-trivial: construction of **inequivalent** topologies for merging off-shell amplitudes to Feynman diagrams.

Solution: count Diagrams w/ a given internal vertex



The number of diagrams in partition (n_1, n_2, n_3) is

$$N(n_1, n_2, n_3) = \frac{(n_1 + n_2 + n_3)!}{S(n_1, n_2, n_3)} \prod_{i=1}^3 \frac{(2n_i - 3)!!}{n_i!}$$

w/ the symmetry factor (**ALPHA** was very inspirational!)

$$S(n_1, n_2, n_3) = \begin{cases} 3! & \text{for } n_1 = n_2 = n_3 \\ 2 \cdot 2 & \text{for } n_3 = 2n_1 = 2n_2 \\ 2 & \text{for } n_1 = n_2 \vee n_2 = n_3 \\ 2 & \text{for } n_1 + n_2 = n_3 \end{cases}$$

The number of all Feynman diagrams is recovered

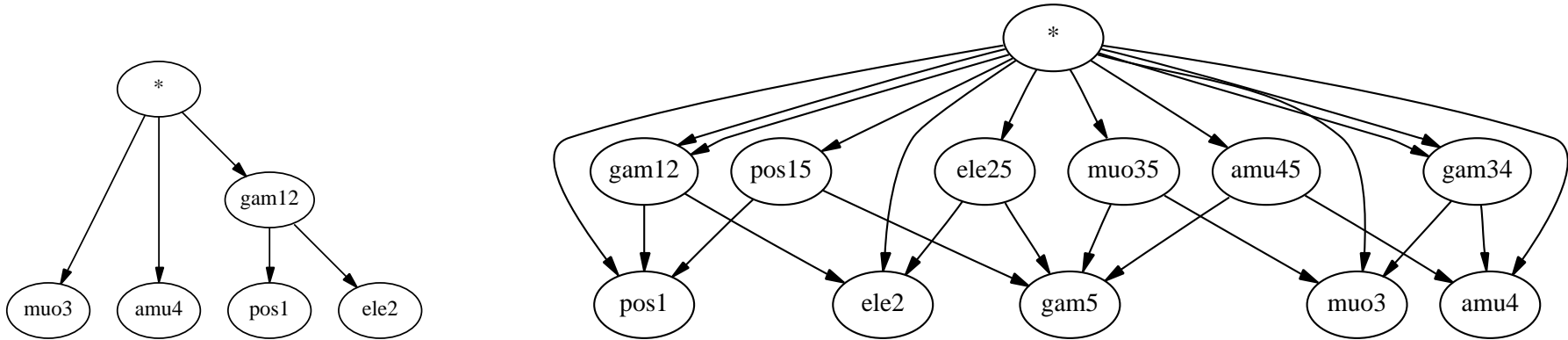
$$\sum_{\substack{n_1 + n_2 + n_3 = n \\ 1 \leq n_1 \leq n_2 \leq n_3 \leq \lfloor n/2 \rfloor}} N(n_1, n_2, n_3) = (2n - 5)!!$$



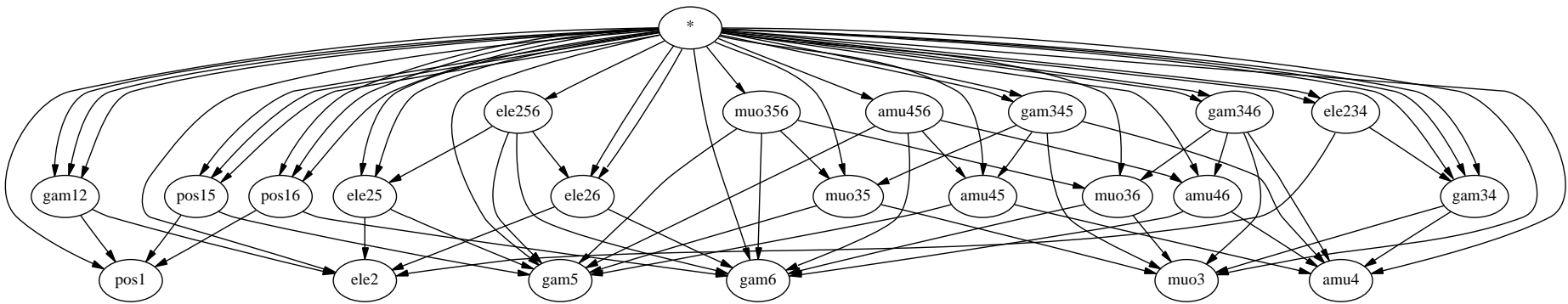
(the funniest formula I have discovered so far ...)

Replace the forest of tree Feynman diagrams by the **Directed Acyclical Graph (DAG)** of the algebraic expression.

- simplest examples: $e^+e^- \rightarrow \mu^+\mu^-$ and $e^+e^- \rightarrow \mu^+\mu^-\gamma$ (only QED)

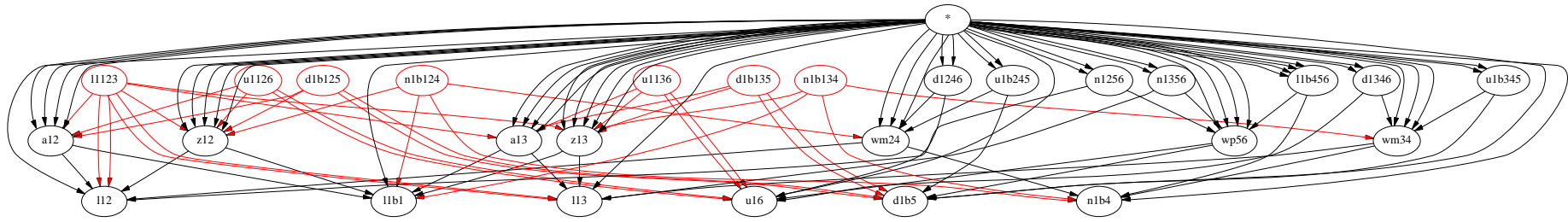


- noticeable savings: $e^+e^- \rightarrow \mu^+\mu^-\gamma\gamma$ (only QED)

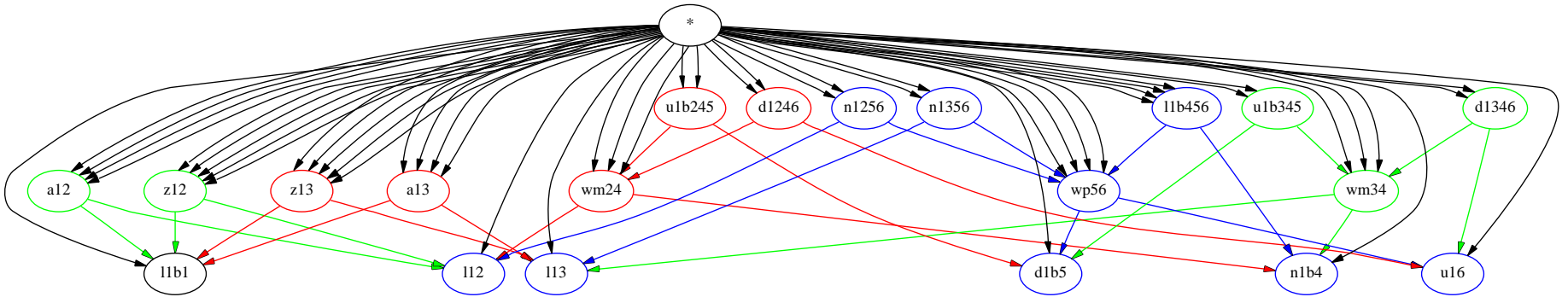


nodes with more than one incoming arrow are calculated only **once!**

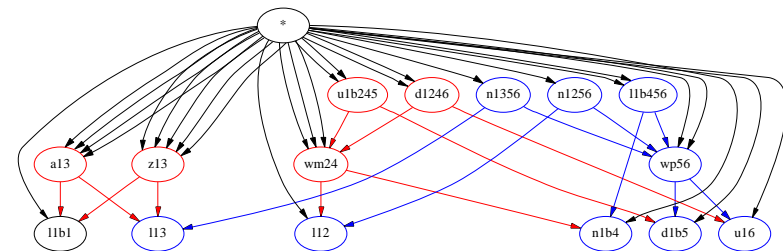
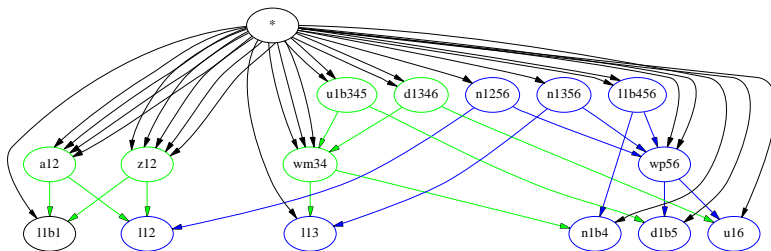
CC20: $e^+ e^- \rightarrow e^+ \nu_e d \bar{u}$:



before and after **eliminating unreachable nodes**:



- two separately gauge invariant **sub-DAGs**, corresponding to the **CC10 grove** and its t-channel complement. NB: these DAGs are **not** disjoint and share the W exchanges $e^-, * \rightarrow \nu_e d \bar{u}$, $e^- \rightarrow \nu_e^* d \bar{u}$, and $\bar{\nu}_e \rightarrow e^+ d \bar{u}$.





✓ 1	DAGs of Feynman Diagrams	2
2	O'Mega	10
	Mission	
	Features	
	Architecture	
	Models	
	Targets	
	Results	
	Shortcomings	
3	Outlook	19



- People: Mauro Moretti, Th. O., Jürgen Reuter, Wolfgang Kilian, ...

Combine the best features of all existing tools for automatic calculation:

- **CompHEP**: comprehensive treatment of cross section & phase space
- **GRACE**: helicity amplitudes
- **MADGRAPH**: convenience & speed
- **Alpha** (& **HELAC**): no factorial growth

and improve on them:

- physics:
 - **systematic** inclusion of physics beyond the standard model: **effective field theories**
 - facilities for consistency checks (**Ward identities**)
 - pave the way for **loops**
- computing:
 - **maintainable** implementation
 - **well defined** extensibility (no wizardry!)



- **optional** instrumentation of the generated code (cf. debugging code of compilers)

- argument checks: momentum, conservation, on-shell condition
- Slavnov-Taylor identities

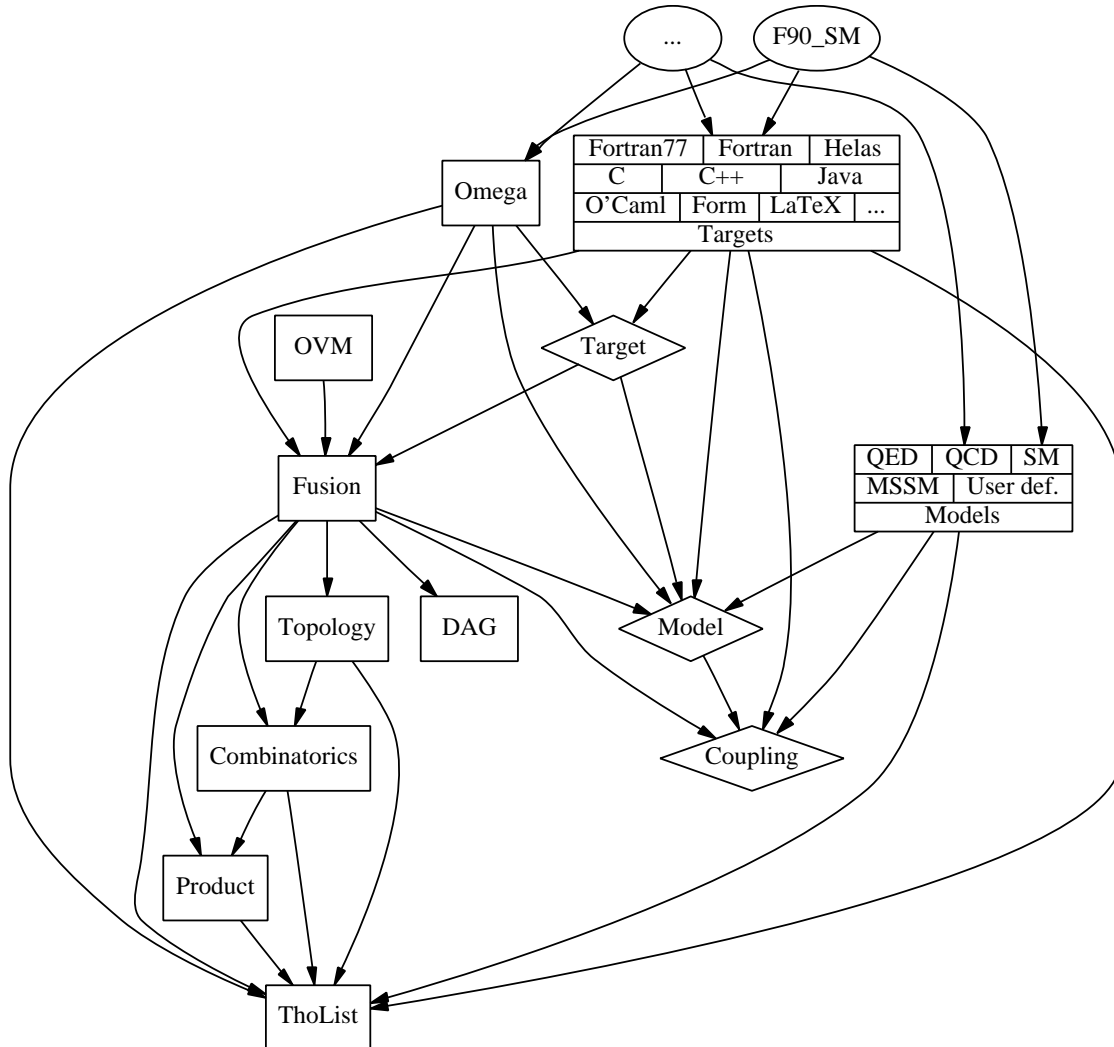
$$\partial_\mu \langle \text{out} | W^\mu(x) | \text{in} \rangle = m_W \langle \text{out} | \phi(x) | \text{in} \rangle$$

still under development

- * universal numerical tolerances hard to find
 - * particularly for tiny helicity violating contributions
- **automatic** recognition of singularities for guiding adaptive Monte Carlo integration
 - analytic **helicity selection rules** speeding up helicity sums for light fermions (not implemented yet)
 - analytic control of **matching** of perturbative, resummed and non-perturbative physics (not implemented yet)



O'Caml's powerful module system supports a very flexible architecture w/ **functors** building applications from independent modules



The module `Targets` contains implementations of the signature `Target` for each target language & the module `Models` contains implementations of `Model` for each (class of) physics models. E.g. the application writing Fortran95 for the standard model is

```
module 0 = Omega.Make(Fusion.Binary)
                (Targets.Fortran) (Models.SM)
let _ = 0.main ()
```



All models must provide the following functions

```
module type T =
  sig
    open Coupling

    type flavor
    val flavors : flavor list
    val flavor_of_string : string -> flavor
    val flavor_to_string : flavor -> string

    val color : flavor -> color
    val lorentz : flavor -> lorentz
    val conjugate : flavor -> flavor

    type gauge
    val propagator : flavor -> gauge propagator
    val goldstone : flavor -> flavor option
    val fermion : flavor -> int

    type constant
    val fuse : flavor -> flavor ->
      (flavor * constant Coupling.t) list
    val parameters : constant parameters
  end
```



For example

$$\mathcal{L}_{CC} = -\frac{g}{2\sqrt{2}} \sum_i \bar{\psi}_i \gamma^\mu (1 - \gamma_5) (T^+ W_\mu^+ + T^- W_\mu^-) \psi_i$$

as O'Caml expression

```
let charged_currents n =  
  [ ((L (-n), Wm, N n), FBF (Psibar, VL, Psi), G_CC);  
    ((N (-n), Wp, L n), FBF (Psibar, VL, Psi), G_CC);  
    ((D (-n), Wm, U n), FBF (Psibar, VL, Psi), G_CC);  
    ((U (-n), Wp, D n), FBF (Psibar, VL, Psi), G_CC) ]
```

```
let triple_gauge =  
  [ ((Ga, Wm, Wp), Gauge_Gauge_Gauge, I_Q_W);  
    ((Z, Wm, Wp), Gauge_Gauge_Gauge, I_G_ZWW) ]
```

used (together with many more) in

```
let vertices =  
  (ThoList.flatmap electromagnetic_currents [1;2;3] @  
   ThoList.flatmap neutral_currents [1;2;3] @  
   ThoList.flatmap charged_currents [1;2;3] @  
   yukawa @ triple_gauge @ quartic_gauge @  
   gauge_higgs @ higgs @ goldstone_vertices)
```

```
let table = F.of_vertices vertices  
let fuse = F.fuse table
```



```
let print_current rhs =
  let wf1 = variable (F.first rhs)
  and wf2 = variable (F.second rhs)
  and p1 = momentum (F.first rhs)
  and p2 = momentum (F.second rhs)
  and vertex, fusion, constant = F.coupling rhs in
  let c = M.constant_symbol constant in
  printf "@, %s " (if F.sign rhs < 0 then "-" else "+");
  begin match vertex with
  | FBF (fb, b, f) ->
      Fermions.print_current (fb, b, f) c wf1 wf2 fusion
  | Gauge_Gauge_Gauge ->
      begin match fusion with
      | F23 | F31 | F12 ->
          printf "g_gg(%s,%s,%s,%s,%s)" c wf1 p1 wf2 p2
      | F32 | F13 | F21 ->
          printf "g_gg(%s,%s,%s,%s,%s)" c wf2 p2 wf1 p1
      end
  end
end
```

Calling Fortran95 functions like

```
pure function g_gg (g, a1, k1, a2, k2) result (a)
  complex(kind=omega_prec), intent(in) :: g
  type(vector), intent(in) :: a1, a2
  type(momentum), intent(in) :: k1, k2
  type(vector) :: a
  a = (0,-1) * g * ((k1-k2)*(a1*a2) &
                  + ((2*k2+k1)*a1)*a2 - a1*((2*k1+k2)*a2))
end function g_gg
```



Radiative corrections to four fermion production (CC20),
minimal standard model in unitarity gauge:

process	Diagrams		O'Mega	
	#	vertices	#prop.	vertices
$e^+ e^- \rightarrow$				
$e^+ \bar{\nu}_e d \bar{u}$	20	80	14	44
$e^+ \bar{\nu}_e d \bar{u} \gamma$	146	730	36	151
$e^+ \bar{\nu}_e d \bar{u} \gamma \gamma$	1112	6672	94	468
$e^+ \bar{\nu}_e d \bar{u} \gamma \gamma \gamma$	12420	86940	168	1246
$e^+ \bar{\nu}_e d \bar{u} \gamma \gamma \gamma \gamma$	138816	1110528	344	3746

Radiative corrections to six fermion production:

process	Diagrams		O'Mega	
	#	vertices	#prop.	vertices
$e^+ e^- \rightarrow$				
$e^+ \bar{\nu}_e d \bar{u} b \bar{b}$	522	3132	58	264
$e^+ \bar{\nu}_e d \bar{u} b \bar{b} \gamma$	4956	34692	108	670
$e^+ \bar{\nu}_e d \bar{u} b \bar{b} \gamma \gamma$	58340	466720	226	2101

- O'Mega amplitudes for up to 7 particles ("2 → 5")
tested against MADGRAPH

😊 agreement for random momenta **always** better than
 10^{-11} (naive Fortran double precision vertices)



😊 **ALPHA** (& **HELAC**): only one program (executable or object file) for each model, can select arbitrary set of external particles (only subject to memory limitations)

😞 **O'Mega** requires a separate function for each amplitude. However:

- for any given study, the set of relevant amplitudes is limited & can be generated by suitable driver programs
- **O'Mega** is **fast**: e. g. 25 seconds for $e^+ \bar{\nu}_e d \bar{u} b \bar{b} \gamma \gamma$ in the full standard model (on my 200 MHz Pentium)
- large incoherent “LHC-style” sums can be build from sums of a few gauge invariant subamplitudes (“groves”, [Boos & Ohl](#))
- can be a blessing in disguise for archival purposes


😞 Fortran Compilers punt at $2 \rightarrow 9$ amplitudes & higher, even though the code is strictly linear

- the fact that no redundancy remains in the code appears to complicate register allocation
- switch to a dedicated virtual machine (probably only a very small loss of performance, see below)



✓ 1	DAGs of Feynman Diagrams	2
✓ 2	O'Mega	10
3	Outlook	19
	Short Term Progress	
	Long Term Progress	



- Comprehensively tested complete standard model in more gauges (R_ξ & axial)
 - interface to **adaptive Monte Carlo VAMP** (see the next talk by Wolfgang Kilian)
 - model independent **deviations from the standard model**
 - ad-hoc: “anomalous” couplings (**don’t forget the TDR ...**)
 - systematically: **effective field theory**
 - **QCD** w/ color factors
 - up to two colored particles are already handled
 - amplitude decomposition for four colored particles appears possible
 - numerical approach to many colored particles
 - **SUSY**
 - **Jürgen Reuter (Darmstadt)** has added unified support for Dirac and Majorana fermions using the Feynman rules of **Ansgar Denner et al.**
-  standard model results already correct
-  *MSSM Lagrangian longa, vita brevis ...*



- **O'Giga: O'Mega Graphical Interface for Generation & Analysis**
 - **not** a replacement or even competition for **JAS (Java Analysis Studio)** or similar tools
 - graphical interface for process selection
 - immediate feedback for parameter changes and simple cuts
- **O'Mega Virtual Machine**
 - ∴ currently, most time is spent in non-trivial vertex evaluations for vectors & spinors, that take $O(10)$ complex multiplications, e. g. $\bar{\Psi}(g_V\gamma_\mu - g_A\gamma_\mu\gamma_5)\Psi$
 - ∴ virtual vertex evaluation machines can challenge native code & avoid compilations
 - ∴ potentially **very** fast as superscalar/parallel **silicon** (or an APE-type machine)
- **O'Tera: O'Mega Tool for Evaluating Renormalized Amplitudes**
 - numerical evaluation of subtracted dispersion relations
 - symbolic & numeric solution of Ward identities