`qgrammar`: A Certified Grammar Compiler for Chiral Gauge Matter

Typed record alphabets, anomaly certificates, repair searches, and UFO back ends for particle-physics grammars

David Elliman
Neuro-Symbolic Ltd
dave@neusym.ai
https://neusym.ai

5 July 2026

Abstract

Two companion papers argued that the Standard Model's elementary interaction rules can be viewed as a typed grammar, and that perturbative amplitudes are naturally evaluated as semiring sums over packed parse forests. This paper turns that viewpoint into a concrete compiler artifact. The public package `qgrammar` defines a typed particle record, a finite response alphabet, vertex productions as typed rewrite rules, exact anomaly-cancellation certificates, a small semiring parser, a minimal repair engine for illegal processes, and a scaffold exporter for the Universal FeynRules Output (UFO) model format. The aim is deliberately modest but useful: not to replace MadGraph, Pythia, Geant4, or other mature high-energy-physics tools, but to provide a certified front end that checks whether a proposed particle grammar is a consistent chiral gauge theory before it is handed to those tools.

The current release certifies one Standard Model generation with a right-handed neutrino: sixteen Weyl states, four anomaly sums that are exactly zero, twenty-nine primitive vertex classes, correct photon and Z read typing, left-handed charged-current typing, sterile-neutrino exclusion from neutral reads, diagram skeletons for $e^+e^- \to \mu^+\mu^-$, and structured repair candidates for illegal signals such as a sterile photon read, $\mu \to e\gamma$, and anomalous mono-photon plus missing-energy events. The paper is written as a numerate graduate-level bridge between the conceptual grammar papers and a reproducible software tool: the mathematics is the type system; the physics certificate is anomaly freedom; the computational object is a compiler; and the back end is the existing HEP simulation stack.

1 Why build a compiler?

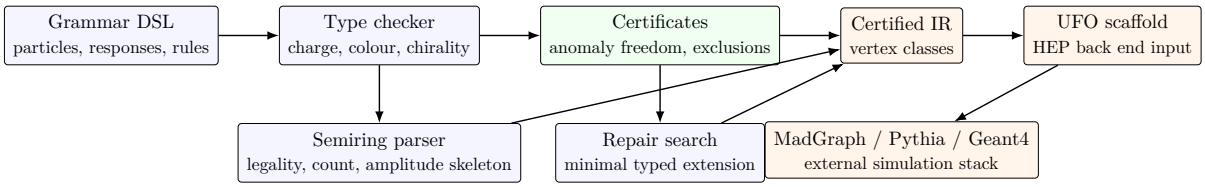
The preceding grammar paper [10] made a structural claim: at tree level the Standard Model's interaction rules can be expressed as a typed grammar over a finite matter alphabet. The semiring sequel [13] then made an evaluation claim: once diagrams are organised as a packed parse forest, legality, diagram counting, amplitude skeletons, and best repairs are all inside evaluations over different semirings [17]. Those papers are conceptual. The question here is more practical:

Can the grammar be made into a useful compiler front end for particle physics?

The answer is yes, at least at the syntax and model-interface level. The public repository [14]

https://github.com/dgedge/grammar

contains the first release of `qgrammar`, a Python command-line tool whose purpose is to make the grammar executable and inspectable. It does not compute precision collider cross sections. It does not try to replace FeynRules [8], UFO [9], MadGraph5_aMC@NLO [4], Pythia [19], or Geant4 [2, 3]. Instead it sits upstream of them:



The compiler framing is useful because it separates three jobs that are often blurred together.

1. **Syntax:** is a proposed process legal under the typed grammar?
2. **Semantics:** if legal, what response weight or amplitude should be attached to its parse forest?
3. **Execution:** how should the resulting model be evaluated numerically by the standard HEP toolchain?

This paper is mainly about the first job and the bridge to the third. The second job is the response layer developed elsewhere [11, 12]. Keeping the boundary explicit is important: a compiler that checks syntax is already useful, but its output is not a precision calculation until a response weight has been supplied.

2 The typed record alphabet

Definition 1 (Particle record)**.** *In `qgrammar` a particle species is represented as a typed record*

$$P = \{\text{generation}, \chi, c, I_3, Y, Q, B, L, s\},$$

where χ is chirality, c is colour representation and triality, I_3 and Y are weak isospin and hypercharge, $Q = I_3 + Y/2$ is electric charge, B and L are baryon and lepton number, and s is a sterile flag.

For one generation the grammar uses the familiar left-Weyl SO(10)-spinor content: a quark doublet, two right-handed quarks, a lepton doublet, a right-handed electron, and a right-handed neutrino [5, 15, 16]. Equivalently, the sixteen Weyl states can be identified with the sixteen codewords of the $[8, 4, 4] = \mathrm{RM}(1, 3)$ record alphabet [10, 18]. The compiler does not need the reader to accept the substrate interpretation in order to be useful: it only needs the typed particle table.

record family	weak type	colour type	neutral reads	compiler role
u_L, d_L	left doublet	triplet	γ, Z (both members charged)	quark doublet
u_R, d_R	right singlet	triplet	γ, Z	right quark singlets
ν_L, e_L	left doublet	singlet	$\nu_L : Z$, $e_L : \gamma, Z$	lepton doublet
ν_R, e_R	right singlet	singlet	ν_R: none, $e_R : \gamma, Z$	sterile state and charged singlet

The phrase "sterile neutrino" is therefore not a label manually pasted onto one particle. It is a derived typing consequence: the right-handed neutrino is the unique codeword with $Q = 0$ and zero Z neutral-current exposure in the primitive grammar.

3 Vertex productions as typed rewrite rules

The grammar treats a Feynman vertex as a typed production,

$$\text{record} \longrightarrow \text{record}' + \text{response}.$$

The response alphabet currently contains the neutral reads γ, Z, charged flips $W^{\pm}$, gluon colour rotations g, the Higgs bridge H, and scalar/gauge self-productions. The rules are ordinary physics rules, but in compiler form.

production class	type condition	example consequence
photon read	record unchanged and $Q \neq 0$	$\nu_R \to \nu_R + \gamma$ is rejected
Z read	nonzero neutral-current exposure	ν_L is Z-active but photon-blind
$W^{\pm}$ flip	left doublet only, I_3 changes by one unit	charged currents are left-handed
gluon read	colour triplet/antitriplet only	leptons are colour-blind
Yukawa bridge	left-right bridge with Higgs response	mass terms are not neutral reads
$\Delta L = 2$ portal	Majorana/recovery channel	leptonic CP can be Majorana-like

Each rule has local checks. For a photon read, for example, the input record and output record must be identical and the electric charge must be nonzero. The compiler prints those checks as certificates. A live run gives

```
LEGAL: e_L -> e_L + gamma
rule:  gamma:  unchanged charged record
- QED electromagnetic read
```

but

```
ILLEGAL: nu_R -> nu_R + gamma
rule:  gamma:  unchanged charged record
- nonzero_electric_charge:  Q(nu_R)=0 .
```

This is a small example, but it shows the intended use. The compiler does not merely say "no". It says which type rule failed.

4 The global certificate: anomaly freedom

Local vertex checks are not enough. A chiral gauge theory must also be globally consistent: its gauge anomalies must cancel [1, 6, 7]. The exact certificate over one generation is

$$\sum_{\text{Weyl}} Y = 0, \tag{1}$$

$$\sum_{\text{Weyl}} Y^3 = 0, \tag{2}$$

$$\sum_{\text{SU(2) doublets}} n_c Y = 0, \tag{3}$$

$$\sum_{\text{SU(3) triplets}} n_w Y = 0, \tag{4}$$

where n_c and n_w are the colour and weak multiplicities of each entry, and the first two sums are the mixed gravitational and cubic U(1) traces respectively. With the convention $Q = I_3 + Y/2$, the one-generation left-Weyl content is

$$\begin{array}{lll} Q_L : (3,2,1/3), & u^c : (\bar{3},1,-4/3), & d^c : (\bar{3},1,2/3), \\ L : (1,2,-1), & e^c : (1,1,2), & \nu^c : (1,1,0). \end{array}$$

Thus

$$\sum Y = 6(1/3) + 3(-4/3) + 3(2/3) + 2(-1) + 2 + 0 = 0, \tag{5}$$

$$\sum Y^3 = 6(1/27) + 3(-64/27) + 3(8/27) + 2(-1) + 8 + 0 = 0, \tag{6}$$

$$\sum_{\mathrm{SU}(2)^2\mathrm{U}(1)} n_c Y = 3(1/3) + 1(-1) = 0, \tag{7}$$

$$\sum_{\mathrm{SU}(3)^2\mathrm{U}(1)} n_w Y = 2(1/3) + (-4/3) + (2/3) = 0. \tag{8}$$

`qgrammar` computes the same result using exact rational arithmetic. The shipped output is:

```
weyl_states = 16
U(1)^3 = 0
grav^2 U(1) = 0
SU(2)^2 U(1) = 0
SU(3)^2 U(1) = 0
ANOMALY CERTIFICATE: PASS
```

Claim 1 (Compiler soundness certificate)**.** *For the supplied Standard Model grammar file, the local type system and the global anomaly certificate agree: the generated primitive vertex classes are local gauge-theory productions, and the one-generation alphabet is a consistent chiral gauge alphabet.*

The important point is not that this is a new anomaly calculation. It is that the calculation becomes a regression test on the grammar. A new particle, a mutated hypercharge, or a proposed repair can be passed through the same certificate before it is promoted to a model.

5 From recognition to semiring parsing

The second grammar paper [13] observed that a Feynman amplitude is not attached to a single parse when several diagrams contribute. It is attached to the parse forest. In semiring-parsing notation [17],

$$\mathrm{Inside}(s) = \bigoplus_{d \in F(s)} \bigotimes_{p \in d} w(p),$$

where $F(s)$ is the packed forest of legal derivations, p ranges over productions in a derivation, and the semiring chooses the meaning:

semiring	output	qgrammar use
Boolean	accept/reject	legality front end
counting	number of parses	diagram skeleton counts
tropical	least edit cost	repair ranking
complex	coherent amplitude skeleton	future response-weight layer

The current compiler includes a small parser for simple sentences. For example,

```
qgrammar parse "e- -> e- | gamma gamma" -derivations
```

(the target record stands to the left of the bar, the ordered response string to its right) returns two derivations, corresponding to left- and right-handed electron records each undergoing two photon reads. In Boolean semantics the sentence is legal; in counting semantics it has two parses; in complex semantics each parse would receive its response weight.

This is still a skeleton calculation. It knows that a parse exists and how many primitive read histories the simplified sentence has. It does not yet compute a loop integral or a precision cross section. That is the correct division of labour: syntax first, response weights second.

6 Minimal grammar repair

The compiler becomes more interesting when the input is illegal. In ordinary model building, an anomalous event is interpreted by proposing new fields and interactions, then checking whether the proposal is consistent. The grammar version asks a smaller, more mechanical question:

> *What is the smallest typed extension that would make this sentence legal while preserving the anomaly and conservation certificates?*

The present repair engine is deliberately simple. It is a ranked catalogue of repair signatures, not a complete automated BSM generator. Even so, it illustrates the right direction. For an anomalous mono-photon plus missing-energy signature it returns, in increasing grammar-edit cost:

1. a sterile neutral state with a Z portal;
2. a dark photon with kinetic mixing and invisible dark records;
3. a leptoquark-like bridge, with warnings about anomaly partners and proton-decay vetoes.

For $\mu \to e\gamma$ it recognises that the one-generation lexer cannot parse the sentence directly and reports charged-lepton flavour violation as an effective dipole repair or a heavy-neutral-lepton loop repair. The phrasing is intentionally conservative: this explains the syntax of the repair, not the branching ratio. A response layer is still needed to compute the rate.

Observation 1 (Repair is model-building as a search problem)**.** *A grammar compiler turns one part of beyond-the-Standard-Model model building into a constrained search: add the smallest new letters and productions that legalise an observed sentence, then reject candidates whose type and anomaly certificates fail.*

7 Back end: certified IR to UFO

The high-energy-physics community already has excellent back ends. A typical chain is

$$\text{model file} \to \text{FeynRules/UFO} \to \text{MadGraph} \to \text{Pythia} \to \text{Geant4}.$$

The weak point is that model files are normally handwritten. The compiler target here is therefore not a rival simulator, but a certified front end.

`qgrammar` currently exports:

- a model intermediate representation (IR);
- a UFO scaffold containing particles, parameters, couplings, Lorentz placeholders, and vertex objects;
- a README inside the UFO export warning that the scaffold is a syntactic/interface artifact, not a precision physical model.

A public smoke test imports the scaffold into MadGraph when MadGraph5_aMC is installed. This is an interface validation, not a physics validation. The next step is to replace placeholder couplings and Lorentz structures by response-derived weights and standard normalisations, then let the existing HEP tools do what they already do well.

8 Using the public artifact

The repository is intentionally lightweight. A typical setup is:

```
git clone https://github.com/dgedge/grammar.git
cd grammar
python -m pip install -e .
```

The minimal checks are:

```
qgrammar anomalies grammar/standard_model.yaml
qgrammar validate
qgrammar check "e- -> e- gamma"
qgrammar check "nu_R -> nu_R gamma"
qgrammar diagrams "e+ e- -> mu+ mu-"
qgrammar repair "mu -> e gamma"
qgrammar export-ir
qgrammar export-ufo /tmp/qgrammar_ufo
```

The 'diagrams' example returns the two expected tree skeletons:

[1] s-channel photon: e+ e- -> gamma* -> mu+ mu-
[2] s-channel Z: e+ e- -> Z* -> mu+ mu-

with certificates that the external charged records are legal, and that generation appears only as an external lexer label.

The end-to-end example script

```
python scripts/end_to_end_examples.py -no-madgraph
```

runs the anomaly certificate, full type-system validation, legal and forbidden process checks, semiring parsing, repair suggestions, diagram skeletons, IR export, and UFO export. If MadGraph is available, the separate backend script runs a UFO import smoke test.

9 What this is, and what it is not

It is worth stating the limits plainly.

- **It is a type checker and compiler scaffold.** It certifies legal primitive productions and exact anomaly cancellation.
- **It is not yet a precision event generator.** Precision cross sections require physical couplings, masses, propagators, loop corrections, PDFs, showering, hadronisation, detector response, and cuts.
- **It does not replace established HEP tools.** It aims to feed them cleaner, checked model input.
- **It does not prove the finite-record substrate.** The substrate interpretation motivates the grammar, but the compiler artifact is useful as a formal model checker even if that deeper interpretation is left aside.

The useful claim is narrower and stronger: a typed grammar front end can make particle-content assumptions explicit, testable, and reproducible. If an observed process is illegal, the compiler can say which rule it violates. If a proposed repair is supplied, the compiler can say which new type obligations and anomaly partners it creates.

10 Next work

The immediate technical programme has five parts.

1. **Full UFO physicalisation.** Replace placeholder Lorentz/coupling entries with response-weighted rules, while retaining the certificate metadata.
2. **Automated anomaly-complete repair.** Extend the current ranked repair catalogue into a finite search over typed extensions.
3. **Generation and mixing.** Add CKM/PMNS labels as response weights, not as syntax-breaking productions.
4. **Loop subtrees.** Treat loop-induced processes ($H \to \gamma\gamma$, $gg \to H$, light-by-light scattering) as composite parse subtrees rather than primitive productions.
5. **Toolchain regression.** Run exported models through the standard collider-simulation path, then compare the compiler's certificates against generated event classes.

The broader research direction is the one suggested by the semiring paper: the S-matrix can be regarded as semiring parsing over a typed field grammar, while measurement probabilities are the response-layer readout of those complex inside scores. `qgrammar` is the first engineering step: it makes the syntax finite, explicit, and executable.

11 Conclusion

The grammar picture stops being only a metaphor once it is executable. `qgrammar` provides a small but concrete artifact: a typed particle record, a Standard Model grammar, exact anomaly certificates, legal and forbidden process checks, semiring parser examples, repair suggestions, diagram skeletons, and a UFO scaffold for existing HEP tools. Its scientific value is not that it proves a new theory by itself. Its value is that it makes assumptions inspectable. A proposed particle alphabet either type-checks or it does not. A proposed vertex either preserves the required records or it does not. A proposed extension either keeps anomaly freedom or it does not. That is the compiler's contribution: it turns part of the grammar of quantum physics into a reproducible object that can be tested, repaired, and handed to the tools physicists already trust.

References

[1] Stephen L. Adler. Axial-vector vertex in spinor electrodynamics. *Physical Review*, 177: 2426–2438, 1969. doi: 10.1103/PhysRev.177.2426.

[2] S. Agostinelli et al. GEANT4—a simulation toolkit. *Nuclear Instruments and Methods in Physics Research Section A*, 506(3):250–303, 2003. doi: 10.1016/S0168-9002(03)01368-8.

[3] J. Allison et al. Recent developments in GEANT4. *Nuclear Instruments and Methods in Physics Research Section A*, 835:186–225, 2016. doi: 10.1016/j.nima.2016.06.125.

[4] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer, H.-S. Shao, T. Stelzer, P. Torrielli, and M. Zaro. The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations. *Journal of High Energy Physics*, 2014(7):79, 2014. doi: 10.1007/JHEP07(2014)079.

[5] John C. Baez and John Huerta. The algebra of grand unified theories. *Bulletin of the American Mathematical Society*, 47(3):483–552, 2010. doi: 10.1090/S0273-0979-10-01294-2.

[6] J. S. Bell and R. Jackiw. A PCAC puzzle: $\pi^0 \to \gamma\gamma$ in the σ-model. *Il Nuovo Cimento A*, 60:47–61, 1969. doi: 10.1007/BF02823296.

[7] C. Bouchiat, J. Iliopoulos, and Ph. Meyer. An anomaly-free version of Weinberg's model. *Physics Letters B*, 38(7):519–523, 1972. doi: 10.1016/0370-2693(72)90532-1.

[8] Neil D. Christensen and Claude Duhr. FeynRules – Feynman rules made easy. *Computer Physics Communications*, 180(9):1614–1641, 2009. doi: 10.1016/j.cpc.2009.02.018.

[9] Céline Degrande, Claude Duhr, Benjamin Fuks, David Grellscheid, Olivier Mattelaer, and Thomas Reiter. UFO – the universal feynrules output. *Computer Physics Communications*, 183(6):1201–1214, 2012. doi: 10.1016/j.cpc.2012.01.022.

[10] David Elliman. The standard model as a certified attribute grammar: Feynman rules as compiler phases on an $[8, 4, 4]$ record alphabet. https://doi.org/10.5281/zenodo.21182033, 2026.

[11] David Elliman. Records and responses: Dressing-blind theorems for monitored quantum instruments. https://doi.org/10.5281/zenodo.21189012, 2026.

[12] David Elliman. Records and responses in the world: A derived fine-structure boundary, certified confinement gaps, and registered discriminators for a finite record substrate. https://doi.org/10.5281/zenodo.21202741, 2026.

[13] David Elliman. Semiring parsing the s-matrix: Packed forests, recursion as dynamic programming, and typed pruning in perturbative field theory. https://doi.org/10.5281/zenodo.21204128, 2026.

[14] David Elliman. `qgrammar`: a certified grammar compiler for chiral gauge matter. https://github.com/dgedge/grammar, 2026. Public GitHub repository, initial public release commit `999f6f8`, accessed 5 July 2026.

[15] Harald Fritzsch and Peter Minkowski. Unified interactions of leptons and hadrons. *Annals of Physics*, 93(1–2):193–266, 1975. doi: 10.1016/0003-4916(75)90211-0.

[16] Howard Georgi. The state of the art—gauge theories. In C. E. Carlson, editor, *Particles and Fields—1974*, volume 23 of *AIP Conference Proceedings*, pages 575–582. American Institute of Physics, 1975. doi: 10.1063/1.2947450.

[17] Joshua Goodman. Semiring parsing. *Computational Linguistics*, 25(4):573–605, 1999.

[18] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland, Amsterdam, 1977.

[19] Torbjörn Sjöstrand, Stefan Ask, Jesper R. Christiansen, Richard Corke, Nishita Desai, Philip Ilten, Stephen Mrenna, Stefan Prestel, Christine O. Rasmussen, and Peter Z. Skands. An introduction to PYTHIA 8.2. *Computer Physics Communications*, 191:159–177, 2015. doi: 10.1016/j.cpc.2015.01.024.