Challenge 03: Celestial CFT Bootstrap

Pure Thought AI Challenge 03

Pure Thought AI Challenges Project

January 18, 2026

Abstract

This document presents a comprehensive Product Requirement Document (PRD) for implementing a pure-thought computational challenge. The problem can be tackled using only symbolic mathematics, exact arithmetic, and fresh code—no experimental data or materials databases required until final verification. All results must be accompanied by machine-checkable certificates.

Contents

Domain: Quantum Gravity Particle Physics
Difficulty: High
Timeline: 6-12 months
Prerequisites: Scattering amplitudes, conformal field theory, Mellin transforms

---

0.1 Problem Statement

0.1.1 Scientific Context

Celestial holography reformulates 4D flat-space quantum gravity as a 2D conformal field theory living on the "celestial sphere" at null infinity. Scattering amplitudes are mapped to correlation functions of operators with conformal weights (,) on the celestial sphere via Mellin transforms.

0.1.2 The Core Question

What is the consistent space of celestial CFTs compatible with graviton scattering amplitudes?

Must satisfy:

- SL(2,) covariance (celestial conformal symmetry)
- Crossing symmetry
- Unitarity (positive norms)
- Weinberg soft graviton theorem
- Regge boundedness at high energies

0.1.3 Why This Matters

- **New holographic paradigm:** Connects 4D gravity to 2D CFT without AdS geometry
- **Rigorous constraints:** Bootstrap approach carves out consistent theories
- **Testable:** Produces islands/no-go regions verifiable by amplitude calculations

---

0.2 Mathematical Formulation

0.2.1 Problem Definition

Celestial amplitude (conformal primary wavefunction):

```
(      , z_i, z _i) =       d      _1 ^{     -1} ...  _n ^{     -1}
 A( _i , z_i, z _i)
```

where:

- A is the usual momentum-space amplitude
- $_i areenergyvariables$

- (zi, zi) are celestial sphere coordinates
- $_iareconformalweights(genericallycomplex)$

Constraints:

- **SL(2,) covariance:**

```
    transforms as  C F T   correlator under z     (az+b)/(cz+d)
```

- **Crossing symmetry:**

Celestial OPE must be associative

- **Unitarity:**

Positive-definite celestial inner product

- **Soft theorem ($\to$ 0):**

```
   (   0   ) ~  S  /   + S  /     + ...
```

where S, S¹ are Weinberg soft factors

- **Regge boundedness:**

```
   (  ) bounded in Regge limit
```

Bootstrap formulation:
Find celestial OPE data C$ijk,$ i satisfying all constraints.
Use crossing equations + positivity to bound or determine OPE coefficients.

0.2.2 Certificate of Correctness

If feasible (consistent celestial CFT found):

- Explicit OPE data: conformal weights i and structure constants Cijk
- Verification: construct celestial correlators, check all constraints
- Cross-check: compute flat-space amplitude, verify it's physical

If infeasible:

- Extremal functional (, z, z) proving no solution exists
- Verification: show applied to crossing equation gives contradiction

0.3 Implementation Approach

0.3.1 Phase 1: Celestial Amplitude Calculator (Months 1-2)

Build Mellin transform engine:

```python
import sympy as sp
from mpmath import mp

mp.dps = 50  # High precision

def mellin_transform(amplitude, omega_vars, delta_vars):
    """
    Compute celestial amplitude via Mellin transform

      (     ) =       d               _i ^{     -1} A(     )
    """
    integrand = amplitude
    for omega, delta in zip(omega_vars, delta_vars):
        integrand *= omega**(delta - 1)

    # Integrate (numerically or symbolically)
    return mp.quad(integrand, [0, mp.inf] * len(omega_vars))
```

Test cases:

- **3-point graviton amplitude:**

```python
def graviton_3pt_amplitude(omega1, omega2, omega3, z1, z2, z3):
    # Momentum conservation:       +       +       = 0 (on
        support)
    return momentum_conserving_delta * kinematic_factor(z1, z2,
        z3)

celestial_3pt = mellin_transform(graviton_3pt_amplitude, ...)
# Should give pure conformal structure
```

- **4-point MHV amplitude:**

Compute celestial transform and verify SL(2,) covariance

0.3.2 Phase 2: Conformal Block Decomposition (Months 2-4)

Celestial conformal blocks:

```python
def celestial_conformal_block(Delta, z, zbar):
    """
    Conformal block for celestial CFT (continuous-spin representation)
    """
    # Use recursion relation or differential equation
    return hypergeometric_solution(Delta, z, zbar)
```

OPE decomposition:

```python
def ope_expansion(celestial_4pt, channel='s'):
    """
    Decompose 4-point function into conformal blocks
```

```

          =   _   C  (   ) G_  (z, z  )
    """
    blocks = [celestial_conformal_block(Delta, z, zbar) for Delta in
        spectrum]
    coefficients = fit_ope_coefficients(celestial_4pt, blocks)
    return {Delta: C for Delta, C in zip(spectrum, coefficients)}
```

0.3.3 Phase 3: Crossing Equations (Months 4-6)

Set up crossing symmetry:

```
def crossing_equation(ope_data_s, ope_data_t):
    """
    Verify/impose s-channel = t-channel OPE

      _s   C_s( s ) G_  s =   _t   C_t( t ) G_  t F_st( t )
    """
    lhs = sum(C_s[Delta] * block_s(Delta) for Delta in ope_data_s)
    rhs = sum(C_t[Delta] * block_t(Delta) * crossing_kernel(Delta)
              for Delta in ope_data_t)

    return lhs - rhs  # Should equal zero
```

Soft theorem constraints:

```
def impose_soft_theorem(ope_data, Delta_soft):
    """
      (   0   ) ~  S  /   + subleading

    Weinberg:  S   = (             )/( z  - z  ) + perms
    """
    # Extract residue at   =0
    residue = extract_residue(ope_data, Delta_soft, pole=1)

    # Check matches Weinberg's soft factor
    assert is_close(residue, weinberg_soft_factor())
```

0.3.4 Phase 4: Bootstrap SDP (Months 6-9)

Formulate optimization:

```
import cvxpy as cp

# Variables: OPE coefficients (continuous spectrum!)
# Discretize    axis
Delta_grid = np.linspace(0.1, 10, num=1000)
C_s = cp.Variable(len(Delta_grid))
C_t = cp.Variable(len(Delta_grid))

# Constraints
constraints = []

# 1. Positivity: |C(  )|       0 (automatic for real C)
constraints.append(C_s >= 0)
constraints.append(C_t >= 0)
```

```

# 2. Crossing symmetry (discretized)
for z_i in z_grid:
    crossing_residual = compute_crossing_residual(C_s, C_t, z_i)
    constraints.append(crossing_residual == 0)  # Or     0

# 3. Soft theorem
constraints.append(enforce_soft_behavior(C_s, C_t))

# Objective: search for extremal functionals or bound OPE data
# (Similar to conformal bootstrap)
```

0.3.5 Phase 5: Extract Results (Months 9-12)

Allowed regions:

```
def scan_celestial_cft_space():
    """
    Scan over assumptions (e.g., minimal  _gap ) and map
    allowed vs. forbidden regions
    """
    results = {}
    for Delta_gap in np.linspace(1, 10, 50):
        try:
            ope_data = solve_bootstrap(Delta_gap)
            results[Delta_gap] = {"status": "feasible", "data":
                ope_data}
        except InfeasibleError as e:
            results[Delta_gap] = {"status": "infeasible",
                "certificate": e.dual}

    return results
```

0.4 Example Starting Prompt

```
I need you to implement the celestial CFT bootstrap from scratch.

GOAL: Determine if a consistent celestial CFT exists for graviton
    scattering
with a minimal conformal weight gap  _gap  = 2.

PHASE 1 - Celestial amplitudes:
1. Write a function that takes a momentum-space amplitude A(     ,  z  )
   and computes its Mellin transform to celestial amplitude  (     ,
      z  ).

2. Implement the 3-point graviton amplitude and compute its celestial
    version.
   Verify it has the right conformal weight structure.

3. Compute the 4-point MHV graviton amplitude's celestial transform.

```

```
PHASE 2 - Conformal blocks:
4. Implement celestial conformal blocks G_  (z, z  ) for continuous-spin
   representations.

5. Decompose the celestial 4-point amplitude into blocks:
         =   _    C  (  ) G_  (z, z  )

PHASE 3 - Crossing & soft theorems:
6. Write down the s-t channel crossing equation.

7. Impose the Weinberg soft graviton theorem as a constraint on    0
   behavior.

PHASE 4 - Bootstrap:
8. Formulate as SDP: find OPE data {C(  )} satisfying crossing +
   positivity
   with          _gap  .

9. Either find a feasible solution OR extract an extremal functional
   proving no solution exists.

Please use high-precision arithmetic and verify all conformal symmetry
transformations explicitly.
```

0.5 Success Criteria

0.5.1 Minimum Viable Result (6 months)

Celestial amplitude machinery working:

- 3-pt and 4-pt amplitudes computed celestially
- SL(2,) covariance verified numerically
- Soft theorems checked

First bootstrap result:

- Crossing equation setup for simple subsector (e.g., MHV only)
- Either: allowed OPE data found, or no-go region certified

0.5.2 Strong Result (9 months)

Multi-channel bootstrap:

- Include all helicity sectors
- Crossing equations in all channels
- Soft+subsubleading soft theorems

Islands/bounds:

- Rigorous allowed region in OPE space
- Or: exclusion of certain conformal weight ranges

0.5.3 Publication-Quality Result (12 months)

Comprehensive classification:

- Full celestial CFT space mapped for graviton scattering
- Phase diagram of consistent theories
- Novel predictions or no-go theorems

Formal verification:

- Certificates exported and verified
- Amplitude calculations cross-checked
- Lean formalization of key results

0.6 Verification Protocol

```python
def verify_celestial_cft(ope_data):
    # 1. Verify SL(2,C) covariance
    for transformation in sl2c_generators():
        transformed = apply_transformation(ope_data,
            transformation)
        assert is_equivalent(transformed, ope_data)

    # 2. Check crossing symmetry
    for z_point in test_points:
        s_channel = ope_sum(ope_data['s'], z_point)
        t_channel = ope_sum(ope_data['t'], z_point)
        assert abs(s_channel - t_channel) < 1e-10

    # 3. Verify soft theorem
    soft_behavior = extract_small_delta(ope_data, Delta=1e-6)
    weinberg = compute_weinberg_soft()
    assert is_close(soft_behavior, weinberg, rtol=1e-8)

    # 4. Unitarity (positive spectral density)
    assert all(c >= 0 for c in ope_data['C_squared'])

    return "VERIFIED"
```

0.7 Milestone Checklist

Mellin transform calculator implemented

3-pt graviton celestial amplitude computed

4-pt MHV celestial amplitude computed

SL(2,) covariance verified

Celestial conformal blocks implemented

OPE decomposition working

Crossing equations formulated

Soft theorem constraints imposed

Bootstrap SDP solver running

First allowed/forbidden result obtained

Extremal functionals extracted (if applicable)

Results formally verified

Next Steps: Begin with implementing Mellin transforms for known tree-level amplitudes and verify conformal covariance before attempting the bootstrap.