Exact CHY Integrand Construction Using Combinatorial Neural Networks and Discrete Optimization

arXiv:2508.02248

Simeng Li
lisimeng1@stu.nxu.edu.cn

School of Physics, Ningxia University

Outline

1. Motivation
2. CHY Formalism and Poles
3. Pole Hierarchy as a Network
4. Discrete Algorithm
5. Examples
6. Conclusion

Motivation

- In amplitude calculations, the number of Feynman diagrams grows fast when n becomes large.
- In CHY-frame, an amplitude is written using a combinatorial integrand.
- Choose the poles (factorization channels) we want, and build the CHY integrand.

Key idea

The pole relations form a fixed message-passing graph (CoNN).
We could solve it with integer updates exactly.

AI+HEP angle: an *exact* solver from an NN architecture

Common ways to exact results

- Large exact linear algebra (big matrices)
- Finite fields + reconstruction
- High-precision floating numbers + truncation

Our route

- Use the CHY pole hierarchy as an NN-like architecture.
- Run integer message passing (forward + backward updates)
- Output integer exponents $K(s_{ij}) \Rightarrow$ exact integrand

CHY Formula

Tree-level n-point amplitude

$$\mathcal{A}_n = \int \frac{\prod_{i=1}^{n} dz_i}{\mathrm{vol}(\mathrm{SL}(2,\mathbb{C}))} \, \Omega(\mathcal{E}) \, \mathcal{I}(z)$$

- Scattering equations: $\mathcal{E}_a = \sum_{b \neq a} \frac{s_{ab}}{z_a - z_b} = 0$.
- Kinematics: $s_{ab} = 2p_a \cdot p_b$ and $s_A = (\sum_{i \in A} p_i)^2$.
- The measure is the same in the CHY formula. The theory choice is in the integrand $\mathcal{I}(z)$.

Integrand Building Blocks

Parke–Taylor factor

$$\mathrm{PT}(\alpha) = \frac{1}{z_{\alpha_1\alpha_2} z_{\alpha_2\alpha_3} \cdots z_{\alpha_n\alpha_1}}$$

- BAS: $\mathrm{PT}(\alpha)\mathrm{PT}(\beta)$
- YM: $\mathrm{PT}(\alpha)\,\mathrm{Pf}'(\Psi)$
- Gravity: $\mathrm{Pf}'(\Psi)\mathrm{Pf}'(\tilde{\Psi})$

Graph rule of thumb: each factor $(z_{ij})^{-1}$ *is a (solid) edge* (i, j).

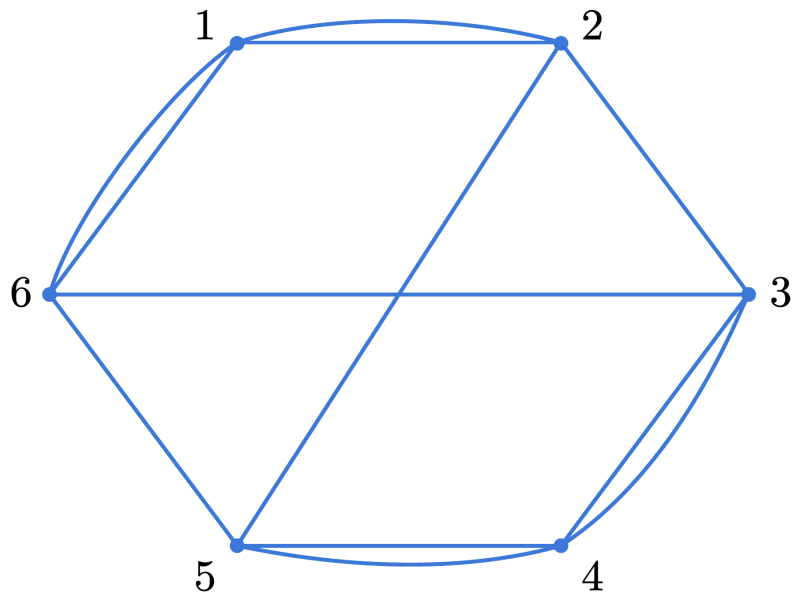


Figure: Graph view of a BAS integrand.

Pole Master Formula

Pole order from graph counting

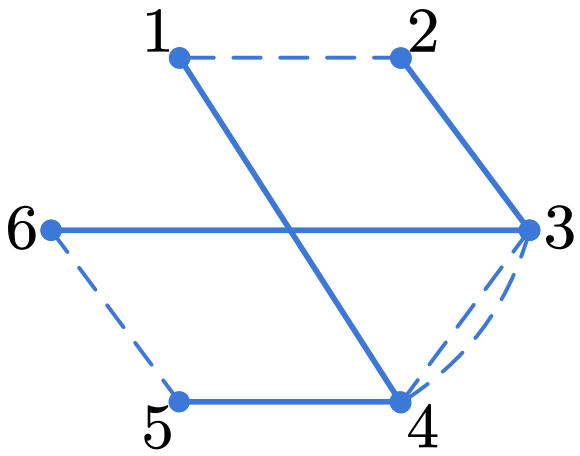
$$\chi(A) = L[A] - 2(|A| - 1)$$

- Here A is a subset of particle labels.
- $L[A]$: number of internal edges in subset A.
- $\chi(A) = 0$ gives a simple pole $1/s_A$ (physical factorization).
- $\chi(A) > 0$ gives a higher-order pole; $\chi(A) < 0$ gives no pole.

Generalized Pole Degree $K(s_A)$

Definition

$$K(s_A) := L_{\text{solid}}[A] - L_{\text{dashed}}[A]$$



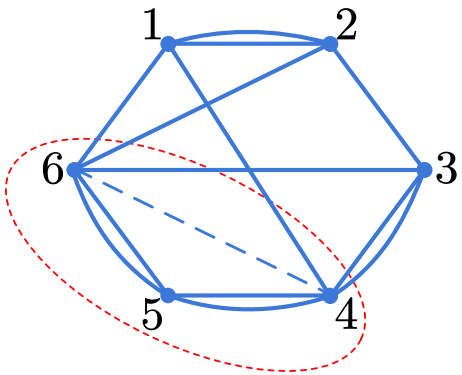
$$I = \frac{z_{12}\, z_{34}^2\, z_{56}}{z_{14}\, z_{23}\, z_{36}\, z_{45}}$$

- Solid edge: factor $(z_{ij})^{-1}$ in the denominator.
- Dashed edge: factor $(z_{ij})^{+1}$ in the numerator.

Generalized Pole Degree $K(s_A)$

Definition

$$K(s_A) := L_{\text{solid}}[A] - L_{\text{dashed}}[A]$$



$$K(s_{456}) = L_{solid}[\{456\}] - L_{dashed}[\{456\}] = 4 - 1 = 3$$

$$\chi(s_{456}) = K(s_{456}) - 2(|\{456\}| - 1) = 3 - 2 \times (3 - 1) = -1$$

- Relation to pole order: $\chi(A) = K(s_A) - 2(|A| - 1)$.
- Converts pole constraints into integer variables.

Recursion and Hierarchy

Recursion

For $|A| \geq 3$,

$$K(s_A) = \frac{1}{|A| - 2} \sum_{\substack{B \subset A \\ |B| = |A| - 1}} K(s_B)$$

- Example: $K(s_{123}) = K(s_{12}) + K(s_{13}) + K(s_{23})$.
- The inputs are the two-particle values $K(s_{ij})$. All higher $K(s_{ij})$ are computed from them by recursion.

Recursion and Hierarchy

- Example: $K(s_{123}) = K(s_{12}) + K(s_{13}) + K(s_{23})$.
- Two-particle values $K(s_{ij})$ are the "bottom-layer" degrees of freedom.

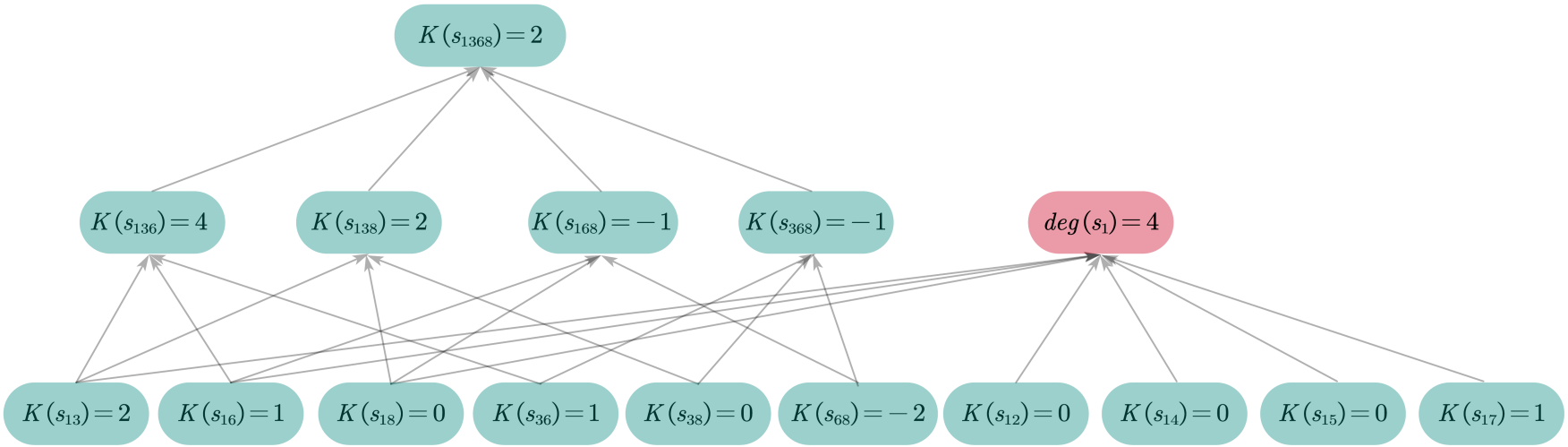
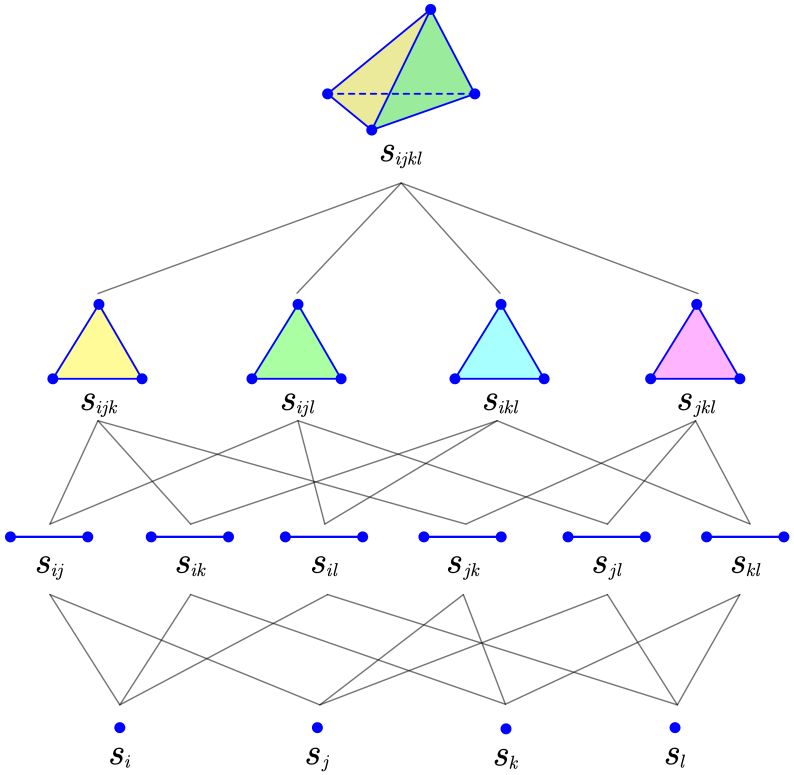


Figure: Hierarchy from the recursion.

Simplicial Complex View

Combinatorial NN

The pole hierarchy is a discrete message-passing network (a combinatorial NN).



- Vertices: single particles.
- Edges: two-particle channels.
- Triangles: three-particle channels, and so on.
- Recursion sends info upward; constraints send it downward.

Inverse Problem

Input and Goal

Input: a desired set of poles $\{s_A\} \Rightarrow$ fixed constraints on $K(s_A)$.
Goal: find an integrand whose K values satisfy all recursions and constraints.

- Feasibility is an integer consistency problem.
- Output integrand (schematic): $\mathcal{I}(z) = \prod_{i<j}(z_{ij})^{-K(s_{ij})}$.

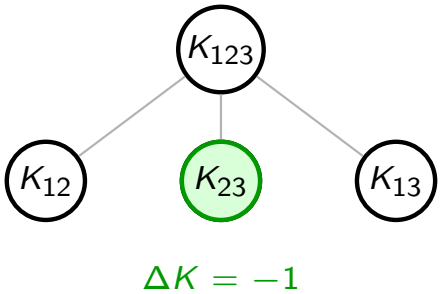
Discrete Backpropagation

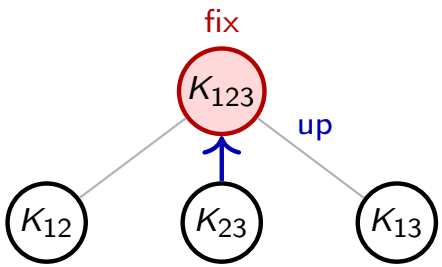
Upward propagation

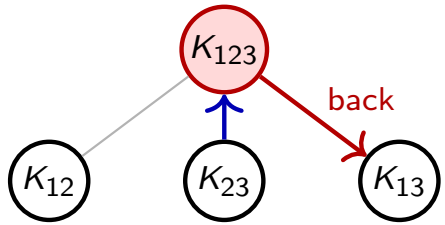
- Modify a lower-level K value.
- Propagate changes to supersets.
- Restore all recursion relations.

Backward propagation

- If propagation hits a fixed constraint,
- redistribute updates to other nodes,
- keep all constraints satisfied.







Algorithm

- Choose a target subset A and set $\Delta K(s_A) = -1$.
- Propagate upward to keep all recursion relations.
- If we hit a fixed node, do a backward update: change other nodes to compensate.
- Stop when all constraints are satisfied.

Key point

Updates are integer-valued and exact.

6-Point Example: Pick Pole s_{12}

Consider a six-point CHY integrand:

$$I_6 = \frac{1}{z_{12}^2 z_{23}^2 z_{34}^2 z_{56}^2 z_{16} z_{45} z_{46} z_{15}}$$

which produces five cubic Feynman diagrams:

$$\frac{1}{s_{12} s_{34} s_{56}} + \frac{1}{s_{123} s_{12} s_{56}} + \frac{1}{s_{123} s_{23} s_{56}} + \frac{1}{s_{156} s_{23} s_{56}} + \frac{1}{s_{156} s_{34} s_{56}}$$

To pick pole s_{12}, we need to retain poles s_{12}, s_{34}, s_{56} and s_{123} while removing poles s_{23} and s_{156}.

6-Point Example: Pick Pole s_{12} - removing poles s_{23}

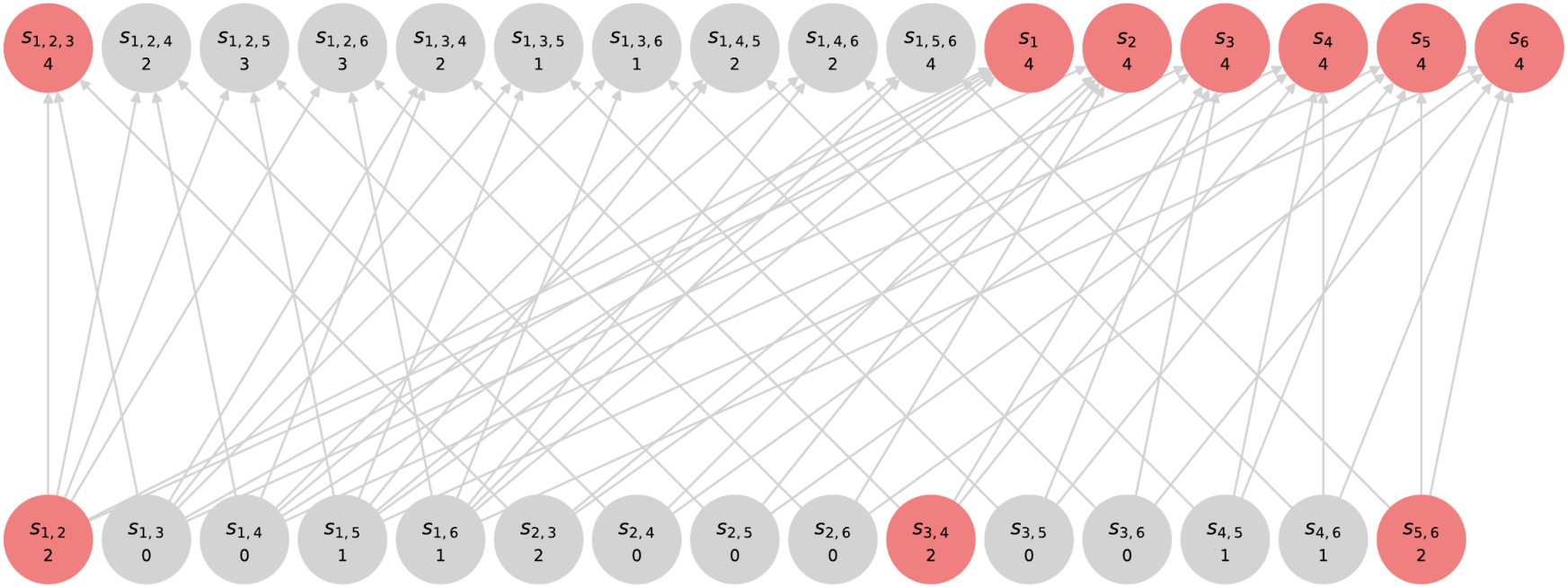


Figure: Initial pole hierarchy (two layers). Bottom: two-particle poles; top: single- and three-particle poles. Gray lines show recursion dependencies. Red nodes are fixed (keep them).

6-Point Example: Pick Pole s_{12} - removing poles s_{23}

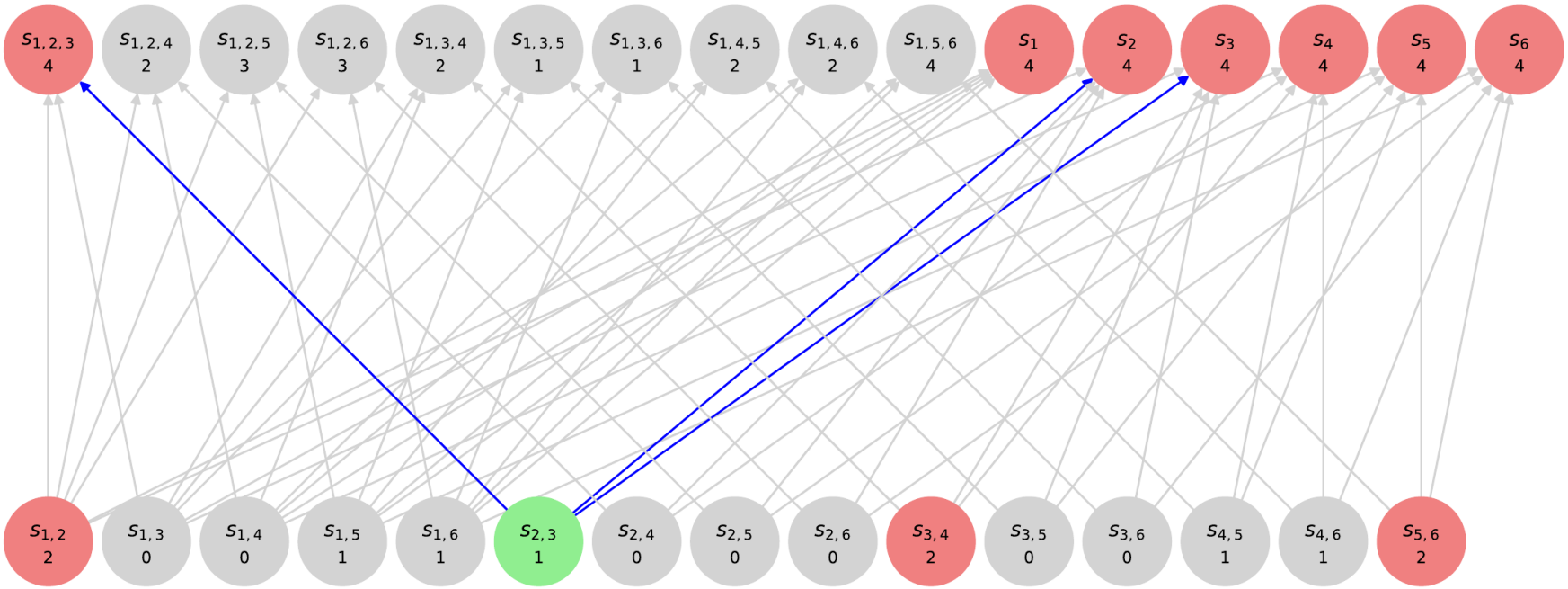


Figure: Step 1 (forward update): set $\Delta K = -1$ at s_{23} (green). Blue arrows show updates to supersets. We hit fixed nodes (red).

6-Point Example: Pick Pole s_{12} - removing poles s_{23}

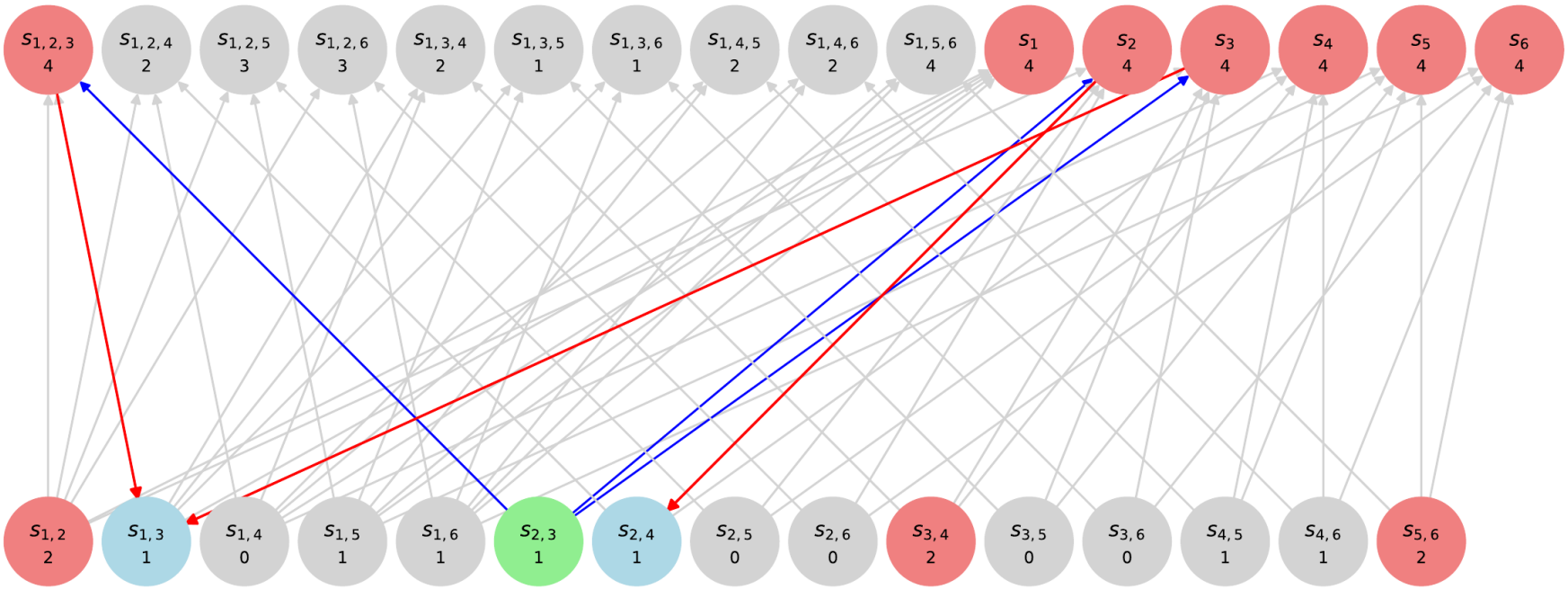


Figure: Step 1 (backward update): when a fixed node is hit, move the change to other nodes (red arrows). Here we increase $K(s_{13})$ and $K(s_{24})$ by $+1$.

6-Point Example: Pick Pole s_{12} - removing poles s_{23}

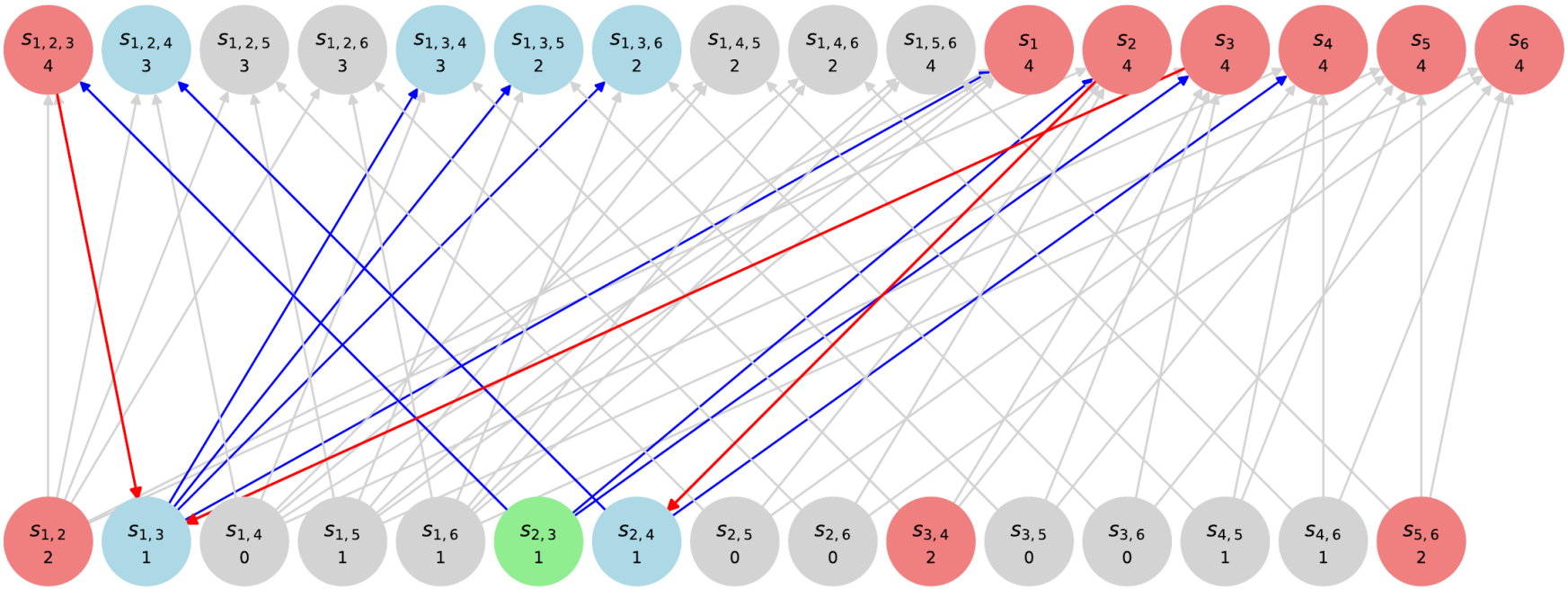


Figure: Step 2 (forward update): propagate the changes from s_{13} and s_{24} upward (blue arrows) and update their three-particle supersets. Fixed nodes are hit again.

6-Point Example: Pick Pole s_{12} - removing poles s_{23}

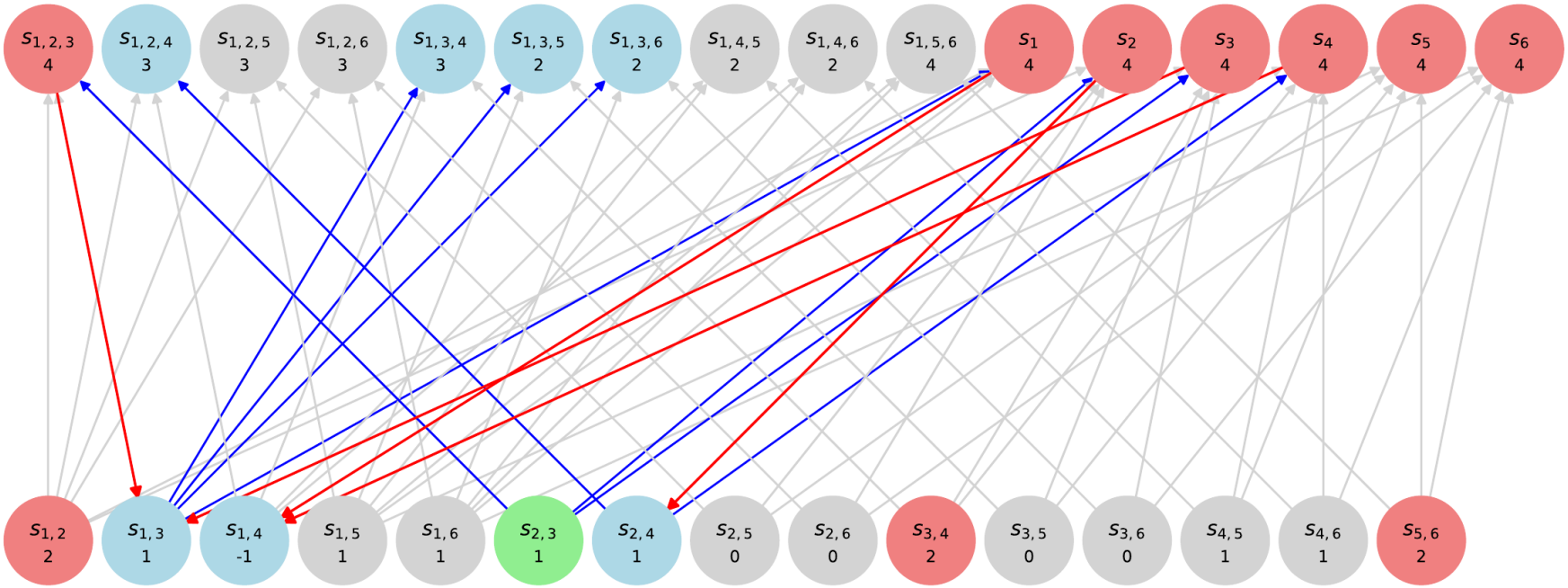


Figure: Step 2 (backward update): adjust $K(s_{14})$ by -1 (red arrows) so all recursion relations stay true while fixed nodes remain unchanged.

6-Point Example: Pick Pole s_{12} - removing poles s_{23}

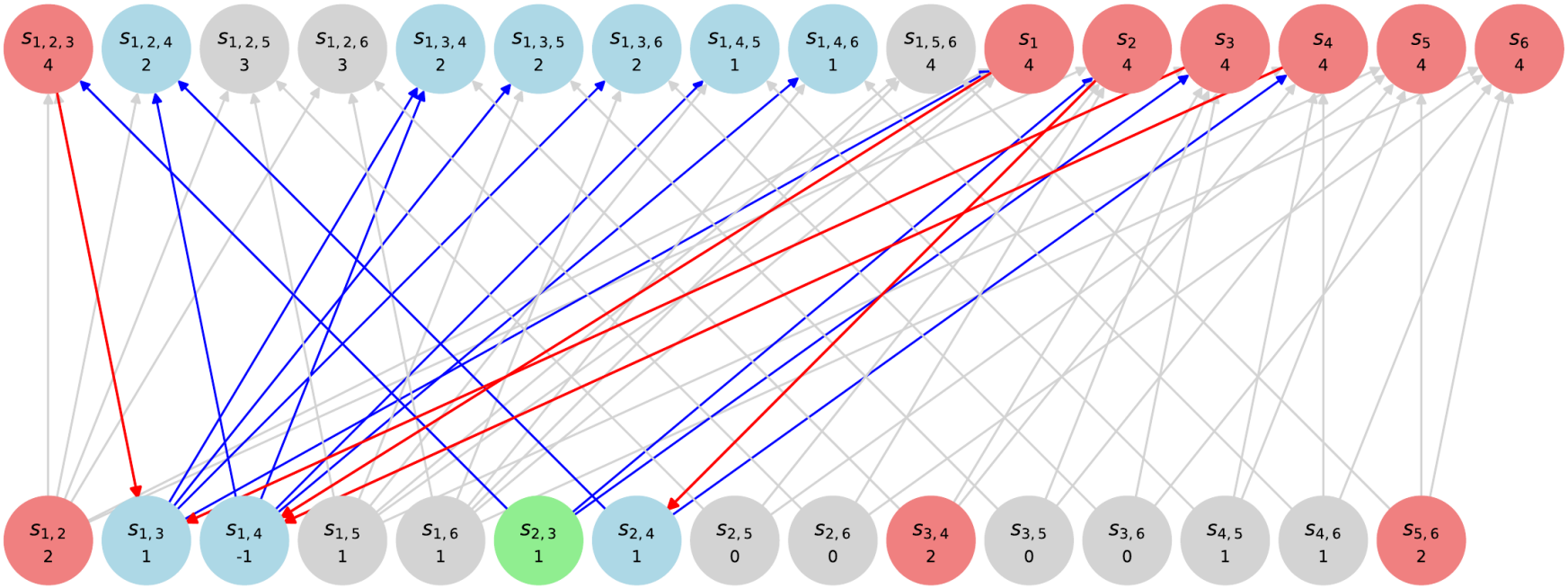


Figure: Step 3 (forward update): propagate once more. Now the pole s_{23} is removed and all constraints are satisfied.

6-Point Example: Pick Pole s_{12} - remove pole s_{156}

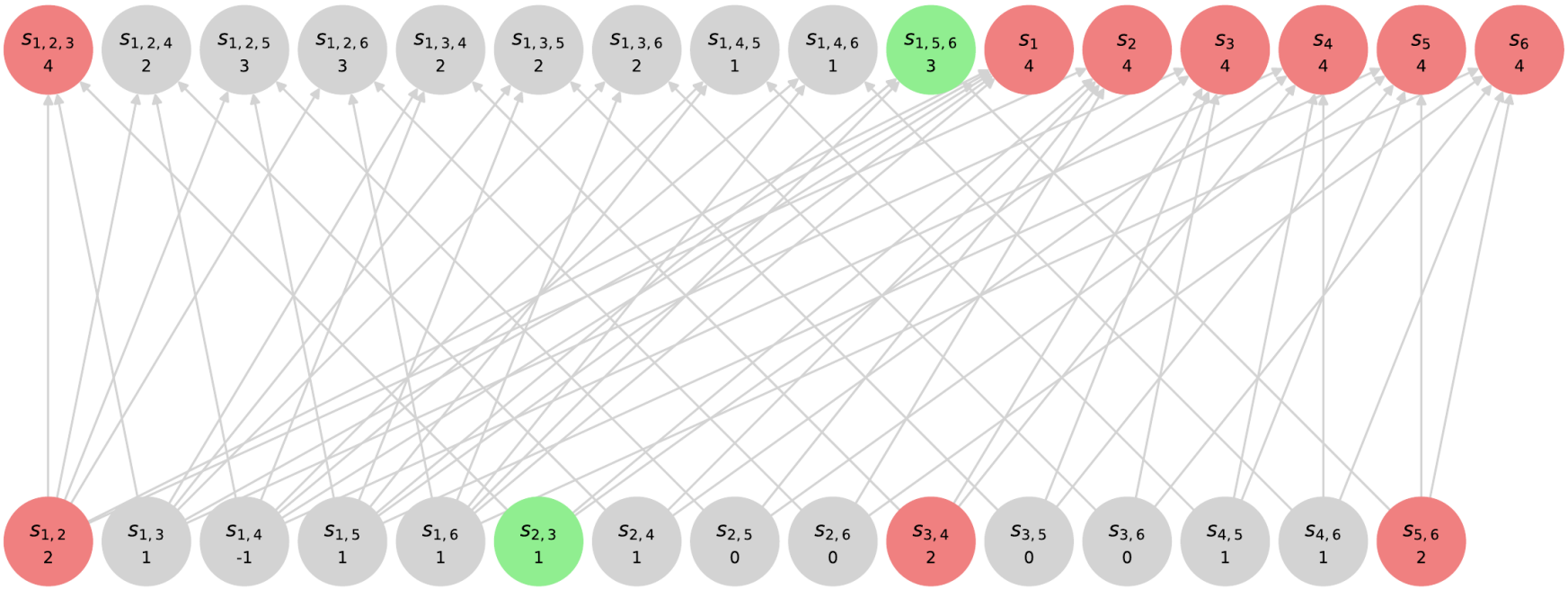


Figure: State after removing s_{23}. Red nodes are fixed. The next target pole is s_{156} (green).

6-Point Example: Pick Pole s_{12} - remove pole s_{156}

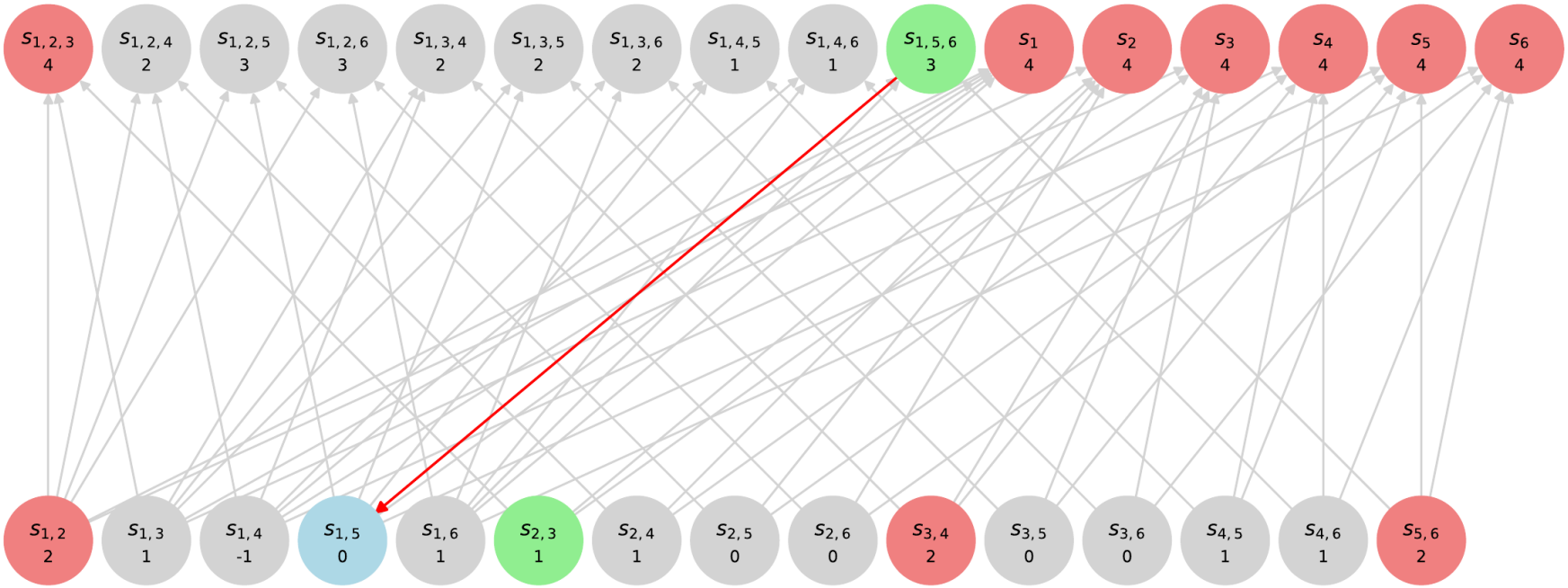


Figure: Remove s_{156}: start with a backward/downward update (red arrow) to the first layer. Choose s_{15} and set $\Delta K = -1$ there.

6-Point Example: Pick Pole s_{12} - remove pole s_{156}

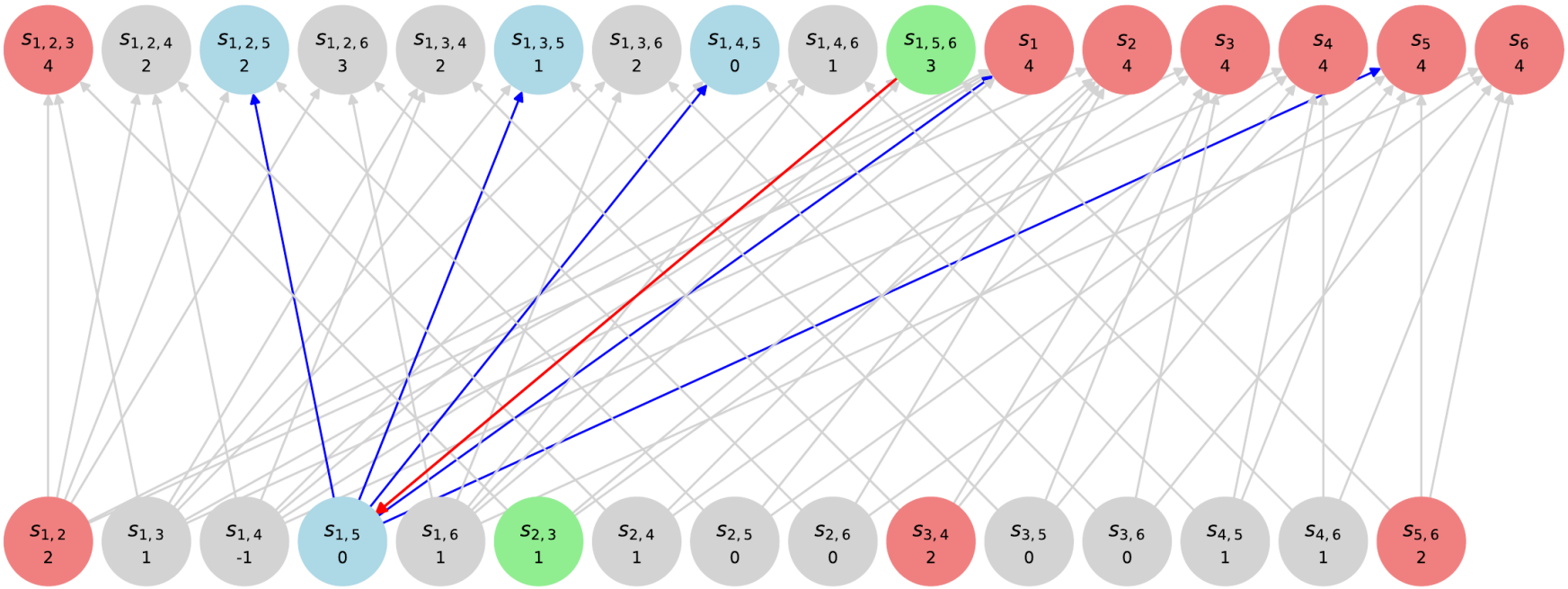


Figure: Forward update from s_{15} (blue arrows): update its three-particle supersets. Fixed nodes s_1 and s_5 are hit (red).

6-Point Example: Pick Pole s_{12} - remove pole s_{156}

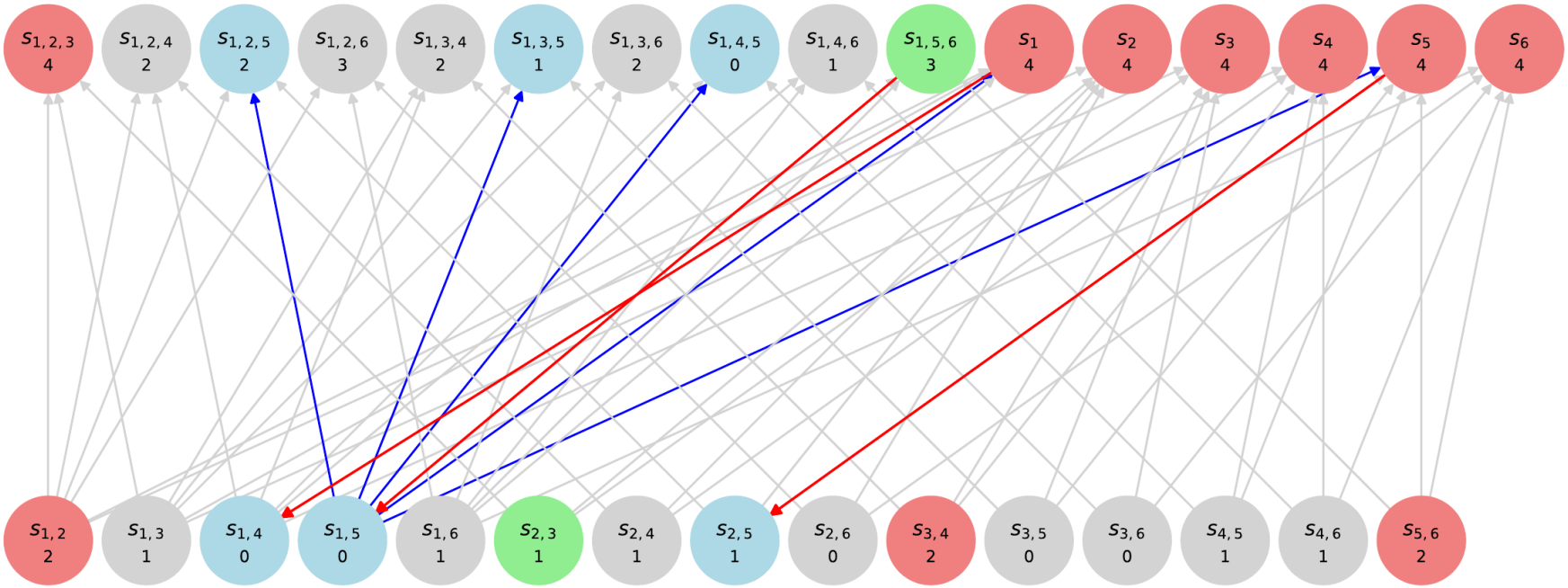


Figure: Backward update (red arrows): increase $K(s_{14})$ and $K(s_{25})$ by $+1$ to keep fixed nodes unchanged.

6-Point Example: Pick Pole s_{12} - remove pole s_{156}

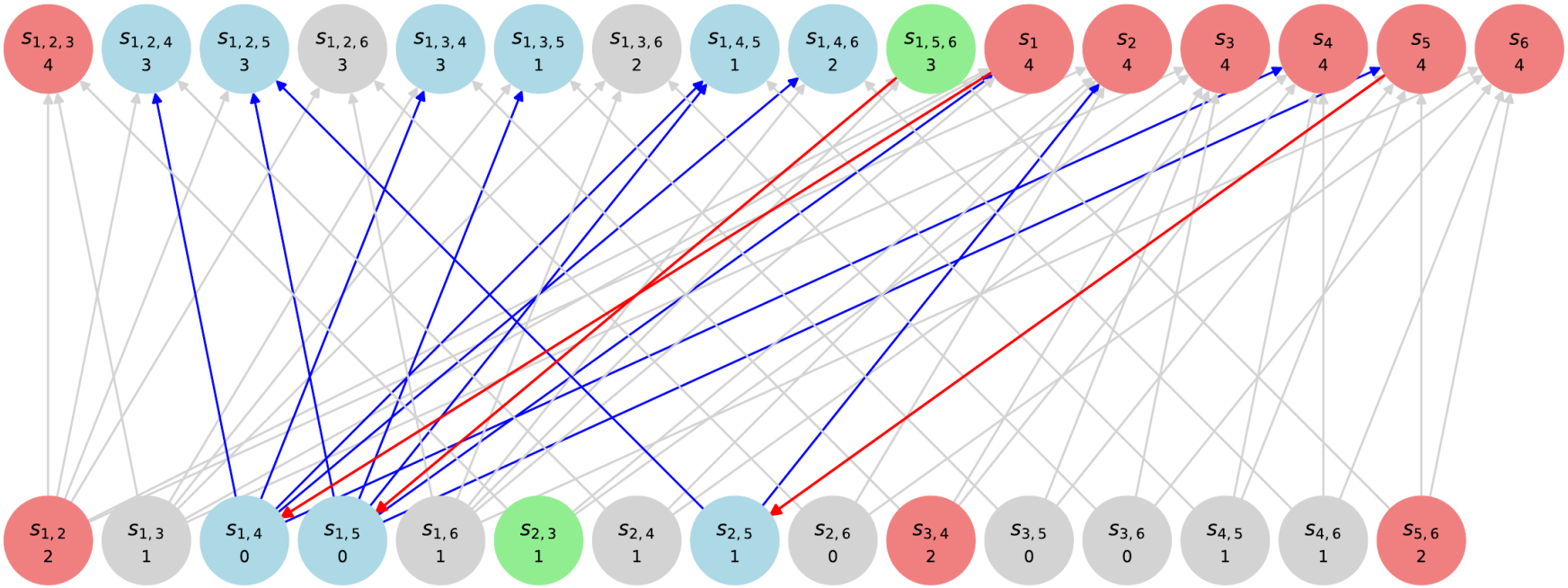


Figure: Forward update (blue arrows): propagate changes from s_{14} and s_{25} to their supersets. Fixed nodes s_2 and s_4 are hit.

6-Point Example: Pick Pole s_{12} - remove pole s_{156}

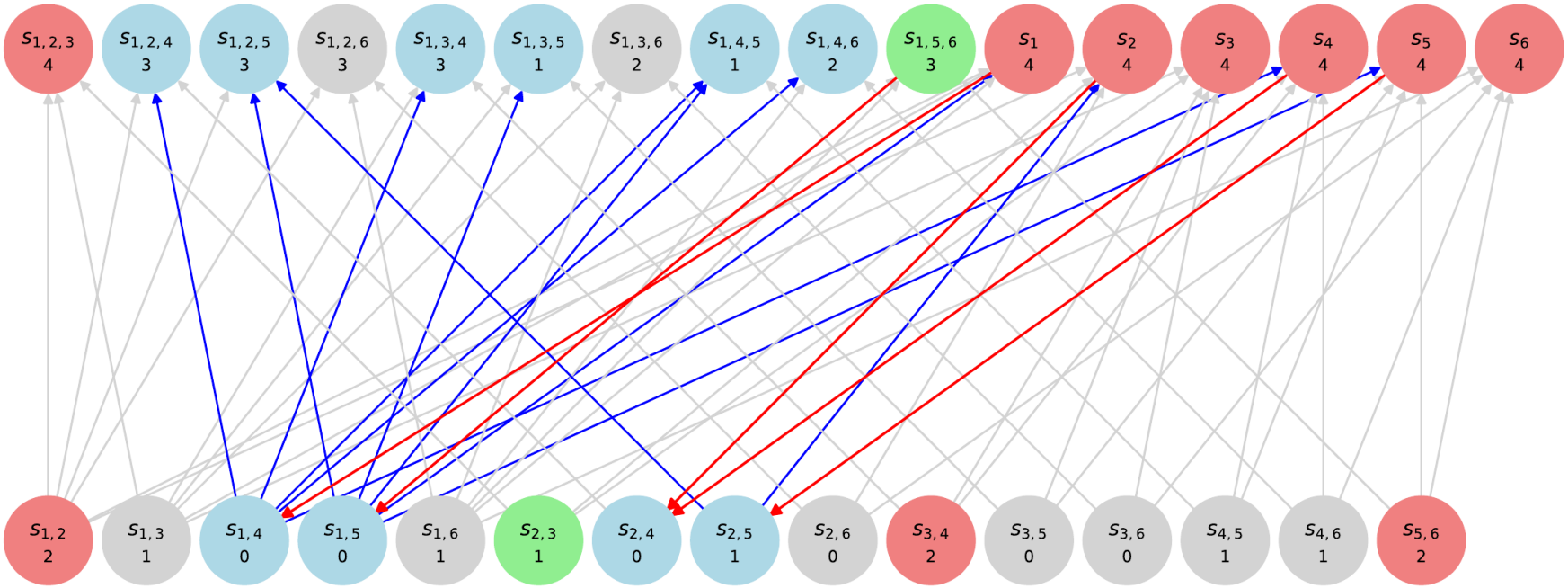


Figure: Backward update (red arrows): decrease $K(s_{24})$ by -1 to resolve the conflict while keeping fixed nodes unchanged.

6-Point Example: Pick Pole s_{12} - remove pole s_{156}

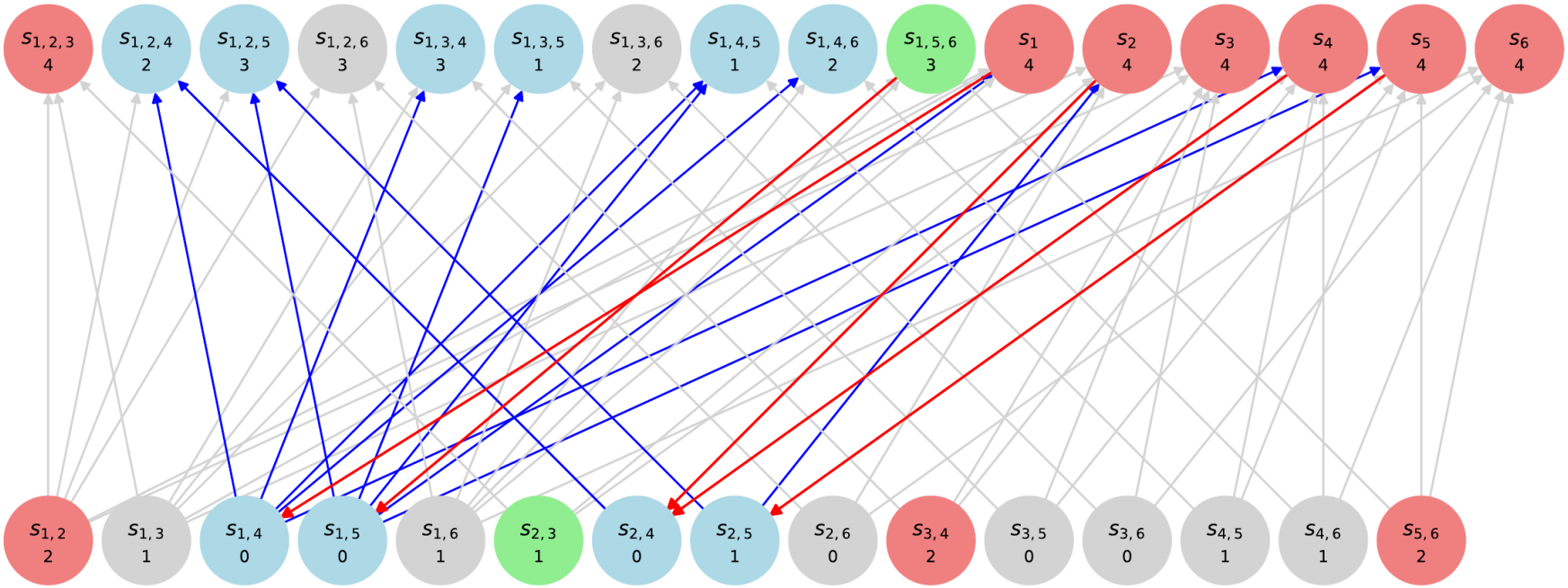


Figure: Final forward update: propagate from s_{24}. Now all constraints are satisfied, and both s_{23} and s_{156} are removed.

6-Point Example: Pick Pole s_{12}

The algorithm stops when all constraints are satisfied.
We obtain the modified CHY integrand:

$$I_6^* = \frac{1}{z_{12}^2 z_{13} z_{16} z_{23} z_{25} z_{34}^2 z_{45} z_{46} z_{56}^2}$$

It gives:

$$\frac{1}{s_{12} s_{34} s_{56}} + \frac{1}{s_{123} s_{12} s_{56}}$$

In this example, both terms contain the desired pole s_{12}.

8-point example (harder case)

Three-Layer Network Structure

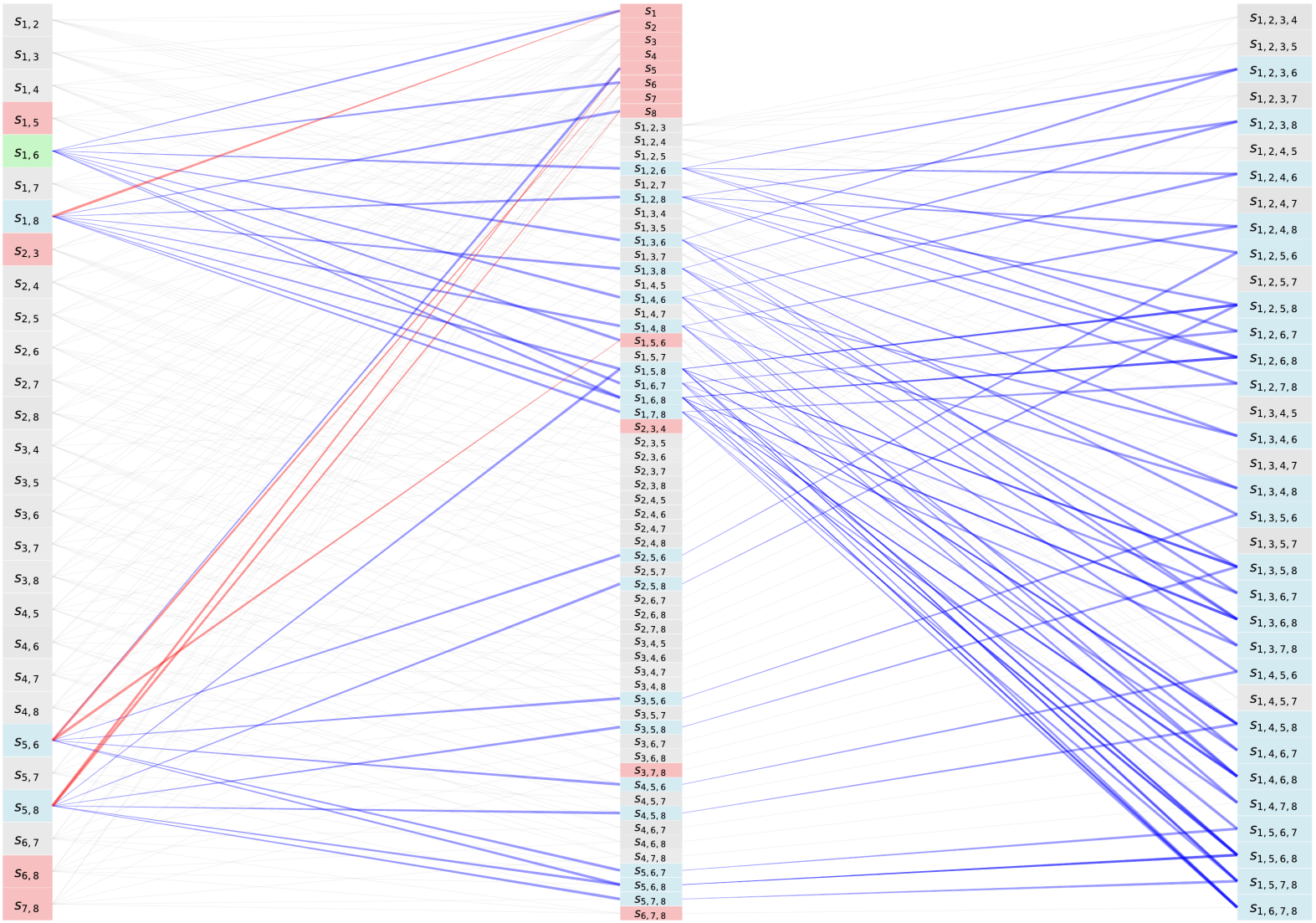
8-point amplitudes require:

- Layer 1: Two-particle poles
- Layer 2: Single- and three-particle poles
- Layer 3: Four-particle poles

What changes at 8 points:

- More constraints
- More forward/backward updates
- The method still finds an integer solution in this example

8-point example (harder case)



Conclusion

1. **CoNN viewpoint:** The CHY pole recursion gives a fixed message-passing graph (simplicial network).
2. **Integer message passing:** Integer message passing is an exact solver on this graph. It builds the integrand and can keep/remove chosen poles.
3. **Outlook:** Extend to more points, and study higher-order poles and loops.

Conclusion

Thank you for your attention!