A max-flow algorithm for positivity of Littlewood-Richardson coefficients

Peter Bürgisser, Christian Ikenmeyer

FPSAC 2009, Hagenberg, Austria DMTCS proc. **AK**, 2009, 265–276

A max-flow algorithm for positivity of Littlewood-Richardson coefficients

Peter Bürgisser[†] and Christian Ikenmeyer[†]

Institute of Mathematics, University of Paderborn, 33098 Paderborn, Germany,
e-mail: {pbuerg,ciken}@math.upb.de

Dedicated to Michael Clausen on the occasion of his 60th birthday.

Abstract. Littlewood-Richardson coefficients are the multiplicities in the tensor product decomposition of two irreducible representations of the general linear group $\mathrm{GL}(n,\mathbb{C})$. They have a wide variety of interpretations in combinatorics, representation theory and geometry. Mulmuley and Sohoni pointed out that it is possible to decide the positivity of Littlewood-Richardson coefficients in polynomial time. This follows by combining the saturation property of Littlewood-Richardson coefficients (shown by Knutson and Tao 1999) with the well-known fact that linear optimization is solvable in polynomial time. We design an explicit *combinatorial* polynomial time algorithm for deciding the positivity of Littlewood-Richardson coefficients. This algorithm is highly adapted to the problem and it is based on ideas from the theory of optimizing flows in networks.

Résumé. Les coefficients de Littlewood-Richardson sont les multiplicités dans la décomposition du produit tensoriel de deux représentations irréductibles du groupe général linéaire $\mathrm{GL}(n,\mathbb{C})$. Ces coefficients ont plusieurs interprétations en combinatoire, en théorie des représentations et en géométrie. Mulmuley et Sohoni ont observé qu'on peut décider si un coefficient de Littlewood-Richardson est positif en temps polynomial. C'est une conséquence de la propriété de saturation des coefficients de Littlewood-Richardson (démontrée par Knutson et Tao en 1999) et le fait bien connu que la programmation linéaire est possible en temps polynomial. Nous décrivons un algorithme *combinatoire* pour décider si un coefficient de Littlewood-Richardson est positif. Cet algorithme est bien adapté au problème et il utilise des idées de la théorie des flots maximaux sur des réseaux.

2000 Mathematical Subject Classification. Primary 05E05. Secondary 90 C27.

Keywords: Littlewood-Richardson coefficients, saturation conjecture, flows in network, polynomial time

1 Introduction

The *Schur polynomials* form a $\mathbb{Z}$-basis of the ring $\Lambda \subseteq \mathbb{Z}[X_1,\ldots,X_n]$ of symmetric polynomials in n variables. They are indexed by partitions λ into at most n parts, which are vectors $\lambda \in \mathbb{N}^n$ of weakly decreasing natural numbers. There are various characterizations of the Schur polynomials s_λ. The shortest one is algebraic and states that $s_\lambda = \Delta^{-1}\det[x_j^{\lambda_i+n-i}]_{1\le i,j\le n}$, where $\Delta = \prod_{i<j}(X_i - X_j)$. A

[†]Partially supported by DFG grant BU 1371/2-1.

combinatorial characterization is $s_\lambda = \sum_T X_1^{\alpha_1(T)} \cdots X_n^{\alpha_n(T)}$, where the sum is over all semistandard tableaux T of shape λ and $\alpha_i(T)$ counts the number of occurences of i in T. We note that s_λ has degree $|\lambda| := \sum_i \lambda_i$. For more information see Stanley (Sta99).

The *Littlewood-Richardson coefficients* $c_{\lambda\mu}^\nu$ are the coefficients in the expansion of the product of two Schur functions in the basis of Schur functions: $s_\lambda s_\mu = \sum_\nu c_{\lambda\mu}^\nu s_\nu$, where the sum is over all partitions ν such that $|\nu| = |\lambda| + |\mu|$. The Littlewood-Richardson coefficients play an important role in various mathematical disciplines (combinatorics, representation theory, algebraic geometry). For instance, they describe the multiplicities in the tensor product decomposition of irreducible representations of the general linear group $\mathrm{GL}(n, \mathbb{C})$. Also, they determine the multiplication in the cohomology ring of the Grassmann varieties.

The well-known Littlewood-Richardson rule provides a combinatorial description of the numbers $c_{\lambda\mu}^\nu$ and leads to several algorithms for computing them, e.g., see (CS84). However, all of these algorithms take exponential time in the size of the input partitions (consisting of integers encoded in binary). Narayanan (Nar06) proved that this is unavoidable: the computation of $c_{\lambda\mu}^\nu$ is a $\#\mathbf{P}$-complete problem. Hence there does not exist a polynomial time algorithm for computing $c_{\lambda\mu}^\nu$ under the widely believed hypothesis $\mathbf{P} \neq \mathbf{NP}$. Surprisingly, as pointed out by Mulmuley and Sohoni (MS05), the positivity of $c_{\lambda\mu}^\nu$ can be decided by a polynomial time algorithm. This can be seen as follows.

Knutson and Tao (KT99) proved the following saturation property: $c_{N\lambda N\mu}^{N\nu} > 0$ implies $c_{\lambda\mu}^\nu > 0$, where N denotes a positive integer. This has implications for various, seemingly unrelated mathematical problems, see Fulton (Ful00). It also has algorithmic consequences: the Littlewood-Richardson rule implies that $\exists N\ c_{N\lambda N\mu}^{N\nu} > 0$ can be rephrased as the feasibility problem of a rational polyhedron. It is well-known that the latter can be solved in polynomial time, cf. (GLS93). Hence by the above saturation property, $c_{\lambda\mu}^\nu > 0$ can be decided in polynomial time.

In (MS05) it was asked whether there is a purely combinatorial algorithm for deciding $c_{\lambda\mu}^\nu > 0$ in polynomial time that does not use linear programming, i.e., one similar to the max-flow or weighted matching problems in combinatorial optimization. The polytopes arising in that setting are integral, i.e. all of its vertices are integral. However, the polytopes occuring in the Littlewood-Richardson situation are not integral, cf. (KTT04).

In this paper we answer the above question in the affirmative by exhibiting a combinatorial polynomial time algorithm for deciding $c_{\lambda\mu}^\nu > 0$. Our algorithm also yields a proof of the saturation property. Knutson, Tao and Woodward (KTW04) proved a conjecture by Fulton stating that $c_{\lambda\mu}^\nu = 1$ iff $c_{N\lambda N\mu}^{N\nu} = 1$ for all N. As a by-product of our developments, we obtain a new proof of this conjecture as well as a combinatorial polynomial time algorithm for deciding multiplicity freeness. (So far this works for strictly descreasing partitions λ, μ, ν only.)

Here is a rough outline of the main ideas underlying our algorithm. By the description in (KT99; Buc00), $c_{\lambda\mu}^\nu$ counts the integral hives with border labels prescribed by λ, μ, ν on the big triangle graph Δ (see §2 for the notation). We establish a bijection between the integral hives and certain integral flows on the dual graph of Δ, which we call *hive flows*. Using this, we convert the problem of deciding the positivity of $c_{\lambda\mu}^\nu$ into the problem of optimizing a certain linear function (the throughput) on the set of integral points of the polyhedron P^b of b-bounded hive flows. We solve this combinatorial optimization problem in analogy to the well-known Ford-Fulkerson algorithm (AMO93) for maximizing flows in networks. We start with the zero flow and iteratively increase the flow f by a fixed integer amount along a cycle while staying in P^b. The set of feasible directions in which to increase can be interpreted as the convex cone of

feasible flows of an auxiliary network $\mathsf{RES}^b(f)$. It is essential and nontrival that one can increase at least by one unit. This key property is expressed in Theorem 11. All of this only works when f is *shattered*, but this nondegeneracy condition is easy to obtain.

In order to obtain a polynomial time algorithm we replace our algorithm by a scaled version that increases flows by integral multiplies of 2^k, but several technical difficulties have to be overcome.

Due to page restrictions in this extended abstract we can only provide sketchs for some of the proofs. The symbol $\square$ at the end of a statement indicates the complete omission of proof. For detailed arguments we refer to the diploma thesis of the second author (Ike08).

Acknowledgment We thank Fritz Eisenbrand for valuable discussions.

2 Preliminaries

2.1 Saturation Property and hive description

Let $\lambda, \mu, \nu \in \mathbb{N}^n$ be partitions such that $|\nu| = |\lambda| + |\mu|$. We start with a triangular array of vertices, $n+1$ on each side, as seen in Figure 1.

This graph is called the *big triangle graph* Δ with vertex set H. To avoid confusion with vertices in other graphs that will be introduced later, vertices in Δ are denoted by underlined capital letters ($\underline{A}$, $\underline{B}$, etc.). The vertices on the border of the big triangle graph form the set B. Denote with $\underline{T}$ the top vertex of Δ and set $H' := H \setminus \{\underline{T}\}$. The graph Δ is subdivided into n^2 small triangles whose corners are graph vertices. We call a triangle in Δ an *upright* triangle if it is of the form '$\triangle$'. Otherwise ('$\triangledown$') we call the triangle an *upside down* triangle. By a *rhombus* $\Diamond(\underline{A}, \underline{B}, \underline{C}, \underline{D})$ with $\underline{A}, \underline{B}, \underline{C}, \underline{D} \in H$ we mean the union of two small triangles next to each other, where $\underline{A}$ is the acute vertex of the upright triangle and $\underline{B}$, $\underline{C}$ and $\underline{D}$ are the other vertices in counterclockwise direction (see Figure 2). Two rhombi are called *overlapping* if they share exactly one triangle.

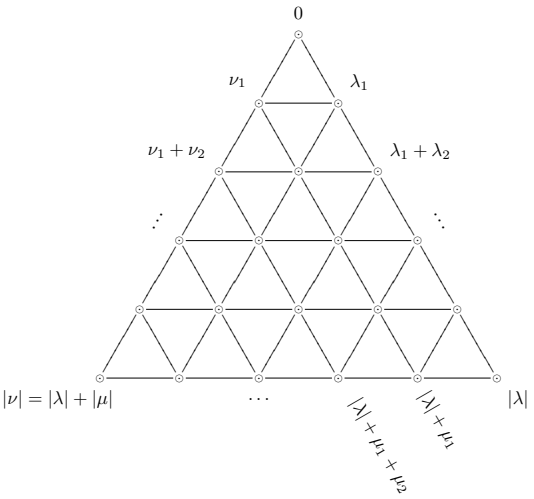

Fig. 1: The big triangle graph Δ with additional border labels resulting from partitions λ, μ and ν, ($n = 5$).

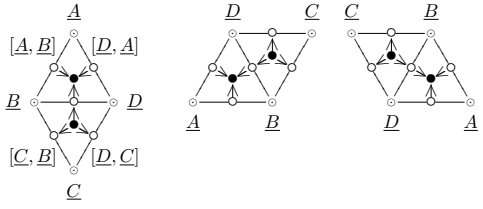

Fig. 2: Rhombus labelings in all possible ways.

Let $h \in \mathbb{R}^H$ be a labeling of the vertices of Δ with real numbers. We call h *integral* iff $h \in \mathbb{Z}^H$. The *h-slack* $\sigma\big(\Diamond, h\big)$ of a rhombus $\Diamond := \Diamond(\underline{A}, \underline{B}, \underline{C}, \underline{D})$ is defined as

$$\sigma\big(\Diamond, h\big) := \big(h(\underline{B}) + h(\underline{D})\big) - \big(h(\underline{A}) + h(\underline{C})\big)$$

(note that $\underline{A}$ and $\underline{C}$ are the acute vertices of $\Diamond$). The rhombus $\Diamond$ is called *h-flat* iff $\sigma\big(\Diamond, h\big) = 0$.

A vertex labeling $h \in \mathbb{R}^H$ is called a *hive* iff the *hive inequalities* $\sigma\big(\Diamond, h\big) \geq 0$ are satisfied for all rhombi $\Diamond$. We note that the set of hives is a convex cone.

For partitions λ, μ and ν with $|\nu| = |\lambda| + |\mu|$, let $b(\lambda, \mu, \nu) \in \mathbb{Z}^B$ denote the border with labels as in Figure 1. This vertex labeling of B is called the *target border* of λ, μ, ν. A border $b \in \mathbb{R}^B$ is called *regular* if for all border vertices $\underline{A}, \underline{B}, \underline{C} \in B$, that are consecutive vertices on the same side of the big hive triangle, we have $b(\underline{A}) + b(\underline{C}) < 2b(\underline{B})$. If λ, μ and ν are strictly decreasing partitions, then the target border $b(\lambda, \mu, \nu)$ is regular.

The following theorem is a consequence of the Littlewood-Richardson rule (KT99; Buc00; PV05).

Theorem 1 *Let λ, μ, ν be three partitions such that $|\nu| = |\lambda| + |\mu|$. Then $c^{\nu}_{\lambda\mu}$ is the number of integral hives with border labels $b(\lambda, \mu, \nu)$.*

2.2 Flows in networks

In this section we introduce basic terminology and facts about flows and augmenting-path algorithms, cf. (AMO93).

Graphs A *graph* $G = (V, E)$ consists of a finite set V of vertices and a finite set $E \subseteq \binom{V}{2}$ of edges whose elements are unordered pairs of distinct vertices. Vertices v and w are called *adjacent* if $\{v, w\} \in E$. We call a vertex v and an edge e *incident* if $v \in e$.

Flows on digraphs Given a graph $G = (V, E)$ we can assign an edge direction to each edge in E by endowing G with an *orientation function* $o\colon E \to V$ that maps each edge to one of its vertices. This turns G into a *directed graph (digraph)*. We call an edge $\{v, w\}$ *directed away from* v (or *directed towards* w) iff $o(\{v, w\}) = v$. The set of edges incident to a vertex $v \in V$ can then be divided into the set $\delta_{\text{in}}(v)$ of edges that are directed towards v and the set $\delta_{\text{out}}(v)$ of edges that are directed away from v. For a mapping $f : E \to \mathbb{R}$ we define

$$\delta_{\text{in}}(v, f) := \sum_{e \in \delta_{\text{in}}(v)} f(e) \quad \text{and} \quad \delta_{\text{out}}(v, f) := \sum_{e \in \delta_{\text{out}}(v)} f(e).$$

Definition 2 (Flow and throughput) A *flow* f on a digraph $G = (V, E, o)$ is a mapping $f\colon E \to \mathbb{R}$ which satisfies $\delta_{\text{in}}(v, f) = \delta_{\text{out}}(v, f)$ for all $v \in V$. We call $\delta(v, f) := \delta_{\text{in}}(v, f)$ the *throughput* of f in v. The flow f is called *integral* iff it only takes integral values.

We note that negative flows on edges are allowed and that therefore the flows on a digraph G form a real vector space $F(G)$.

Capacities We can assign capacities to a digraph $G = (V, E, o)$ by defining two functions $u\colon E \to [0, \infty], e \mapsto u_e$ and $l\colon E \to [-\infty, 0], e \mapsto l_e$, which we call the *upper bound* and *lower bound*, respectively. A digraph with capacities is sometimes called a *network* in the literature. We will tacitly assume that $u_e = \infty$ and $l_e = -\infty$ if no other requirement for the edge e is made.

Definition 3 (Feasible flow) Let $G = (V, E, o)$ be a digraph with capacities u and l. A flow f on G is said to be *feasible with respect to u and l* if $l_e \le f(e) \le u_e$ for each edge $e \in E$. The set $P_{\text{feas}}(G) \subseteq F(G)$ of feasible flows on G is said to be the *polyhedron of feasible flows on G*.

Definition 4 (Cycle) (1) A *cycle* $c = (v_1, \ldots, v_\ell, v_{\ell+1} = v_1)$ on a graph $G = (V, E)$ is a finite sequence of at least 3 vertices in V in which for all $1 \le i < j \le \ell$ we have that $v_i \ne v_j$ and for all $1 \le i \le \ell$ we

have $\{v_i, v_{i+1}\} \in E$. We call $e = \{v_i, v_{i+1}\}$ the edges of c and write $e \in c$. The length $\ell(c) := \ell$ of c is defined as the number of edges in c.

(2) Suppose an orientation function o is fixed. We call $e = \{v_i, v_{i+1}\}$ a forward edge of c iff e is directed away from v_i, and a backwards edge of c otherwise. The cycle c is called *well-directed* iff all of its forward edges e satisfy $u(e) > 0$ and all of its backward edges e satisfy $l(e) < 0$.

(3) We assign to c the *cycle flow* f_c by setting $f_c(e) = 1$ for its forward edges e and $f_c(e) = -1$ for its backward edges e. All other edges carry flow zero.

To simplify notation, we will identify a cycle c with its cycle flow f_c. We remark that c is well-directed iff there is an $\varepsilon > 0$ such that εc is a feasible flow.

It is well-known and easy to see that feasible flows can be decomposed into cycles as follows.

Lemma 5 (Flow decomposition) *Given a digraph $G = (V, E, o)$ and a flow f on G. Then there exist cycles $c_1, \ldots, c_m$ on G, $m \leq |E|$, and $\alpha_1, \ldots, \alpha_m \in \mathbb{R}_{>0}$ such that $f = \sum_{i=1}^{m} \alpha_i c_i$ and for all i and all $e \in c_i$ we have $\operatorname{sgn}(f(e)) = \operatorname{sgn}(c_i(e))$. We call α_i the* multiplicity *of the cycle c_i in the decomposition. Moreover, if f is feasible, then $c_1, \ldots, c_m$ are well-directed.* □

3 Hives and flows

We transfer the problem of finding an integral hive into the language of flows.

The graph structure We now define a bipartite planar digraph $G = (V, E, o)$, which is essentially the dual graph of Δ. The definition is similar to one in (Buc00): G has one fat black vertex in the middle of each small triangle of Δ. In addition there is one circle vertex on every triangle side (see Figure 3). We denote a circle vertex between two vertices $\underline{A}$ and $\underline{B}$ of upright triangles (read in counterclockwise direction) as $[\underline{A}, \underline{B}]$. Each fat black vertex is adjacent to the three circle vertices on the sides of its triangle. There is an additional fat black vertex o with edges from o to all circle vertices that lie on the border of the big triangle. The graph G is embedded in the plane in a way such that the top vertex $\underline{T}$ lies in the outer face, where a *face* is a region bounded by edges, including the outer, infinitely-large region.

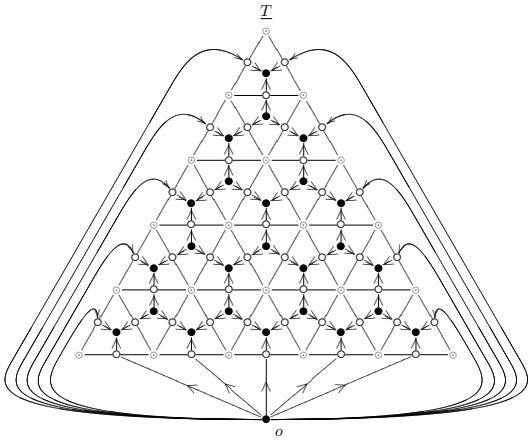

Fig. 3: The digraph G and graph Δ.

Next we assign a direction to each edge in G (see Figure 3). The edges incident to o are directed from o towards the border of the big triangle graph. The edges in an upright triangle are directed towards the incident fat black vertex, while the edges in an upside down triangle are directed towards the incident circle vertex.

Let $F = F(G)$ denote the vector space of flows on G. Note that a flow f on G is completely defined by its throughput $\delta([\underline{A}, \underline{B}], f)$ on each circle vertex $[\underline{A}, \underline{B}]$.

Winding numbers Let $\underline{A} \in H$. We define $\mathscr{NW}(\underline{A})$ to be the set of circle vertices in V that lie on the northwest diagonal drawn from $\underline{A}$. This diagonal hits a border vertex $\underline{B} \in B$. Define $\mathscr{NE}(\underline{A})$ to be the set of circle vertices in V that lie on the northeast diagonal drawn from that border vertex $\underline{B}$. Now define the *winding number* of a vertex $\underline{A} \in H$ with respect to a flow $f \in F$ as

$$\text{wind}(\underline{A}, f) = \sum_{v \in \mathscr{NW}(\underline{A})} \delta(v, f) - \sum_{v \in \mathscr{NE}(\underline{A})} \delta(v, f).$$

The winding number is linear in the flow f. We note that for a cycle flow f_c, this coincides with the familiar topological notion of the winding number of the cycle c with respect to the point $\underline{A}$.

Lemma 6 *There is an isomorphism $\eta : \mathbb{R}^{H'} \to F$ between the real vector space $\mathbb{R}^{H'}$ of vertex labels in Δ, in which the top vertex $\underline{T}$ has value 0, and the real vector space F of flows on G. For $h \in \mathbb{R}^{H'}$ the flow $\eta(h)$ is defined by requiring $\delta([\underline{A}, \underline{B}], \eta(h)) = h(\underline{A}) - h(\underline{B})$. The inverse of η is given by $\eta^{-1}(f)(\underline{A}) = wind(\underline{A}, f)$ for $f \in F$. Both η and η^{-1} preserve integrality.* □

Hive inequalities on flows As η is an isomorphism, we can identify a flow $f \in F$ with its vertex labeling $h = \eta^{-1}(f) \in \mathbb{R}^{H'}$. We define the *$f$-slack* of a rhombus $\Diamond = \Diamond(\underline{A}, \underline{B}, \underline{C}, \underline{D})$ as $\sigma\big(\Diamond, f\big) := \sigma\big(\Diamond, h\big)$. Note that $\sigma\big(\Diamond, f\big) = \delta([\underline{D}, \underline{C}], f) - \delta([\underline{A}, \underline{B}], f) = \delta([\underline{D}, \underline{A}], f) - \delta([\underline{C}, \underline{B}], f)$. Thus the hive inequalities $\sigma\big(\Diamond, h\big) \geq 0$ translate into the following simpler linear inequalities

$$\delta([\underline{A}, \underline{B}], f) \leq \delta([\underline{D}, \underline{C}], f). \tag{1}$$

We call f a *hive flow* if $\eta^{-1}(f)$ is a hive. Similarly, we speak of f-flat rhombi.

4 The algorithmic idea

We now introduce the optimization problem to be solved for deciding whether a Littlewood-Richardson coefficient is positive.

Define the set $\mathscr{S} \subset V$ of *source vertices* as the set of all circle border vertices of G lying on the right or bottom border of the big triangle. Define the set $\mathscr{T} \subset V$ of *sink vertices* as the set of all circle border vertices of G lying on the left border of the big triangle. We call $\delta(f) := \sum_{s \in \mathscr{S}} \delta(s, f)$ the (global) *throughput* of f. Note that $\delta \colon F \to \mathbb{R}, f \mapsto \delta(f)$ is a linear map.

For a given border vertex labeling $b \in \mathbb{R}^B$ we define now the network G^b on the digraph G by introducing the capacities $u_{\{o,s\}} := b(\underline{A}) - b(\underline{B})$ for all $s = [\underline{A}, \underline{B}] \in \mathscr{S}$ and $l_{\{o,t\}} := b(\underline{A}) - b(\underline{B})$ for all $t = [\underline{A}, \underline{B}] \in \mathscr{T}$. We call the feasible flows of G^b b-bounded. The set of b-bounded hive flows is a polyhedron that will be denoted by P^b.

We call an edge $\{o, s\}$ *used to capacity* with respect to a flow $f \in P^b$ iff $\delta(s, f) = u_{\{o,s\}}$. Similarly, we say that the edge $\{o, t\}$ is *used to capacity* with respect to f iff $\delta(t, f) = l_{\{o,t\}}$.

The following lemma shows the significance of the *polyhedron P^b of b-bounded hive flows*.

Lemma 7 *Let $b = b(\lambda, \mu, \nu)$ be the target border of partitions λ, μ and ν with $|\nu| = |\lambda| + |\mu|$. Then*

(1) For all $f \in P^b$ we have $\delta(f) \leq |\nu|$.

(2) $c^{\nu}_{\lambda\mu}$ equals the number of integral $f \in P^b$ such that $\delta(f) = |\nu|$.

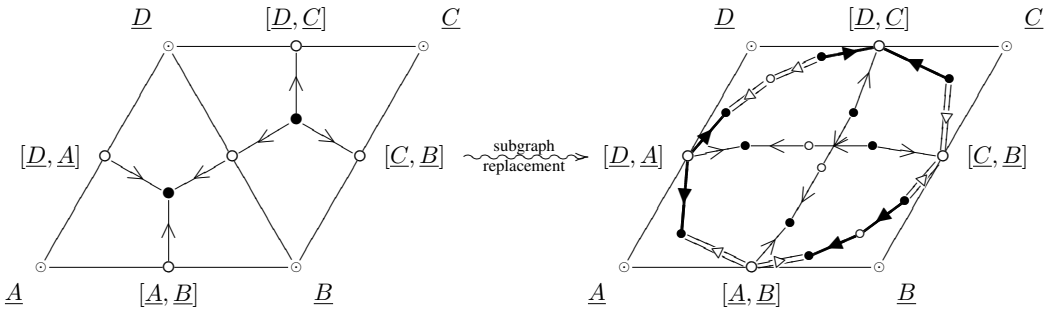

Fig. 4: The subgraph replacement for an f-flat rhombus $\Diamond(\underline{A}, \underline{B}, \underline{C}, \underline{D})$.

Proof: (1) We have $\delta(f) = \sum_{s \in \mathscr{S}} \delta(s, f) \leq |\lambda| + |\mu| = |\nu|$.

(2) According to Theorem 1, $c_{\lambda\mu}^{\nu}$ equals the number of integral hives with border labels b. Lemma 6 shows that this number equals the number of b-bounded integral hive flows with throughput $|\nu|$. □

Lemma 7 translates the problem of deciding positivity of Littlewood-Richardson coefficients to the problem of optimizing the linear function δ on the integer points of the polyhedron P^b. We will solve this combinatorial optimization problem in analogy to the well-known Ford-Fulkerson algorithm (AMO93) for maximizing flows in networks. We start with the zero flow and iteratively increase the flow f by one unit along a cycle while staying in P^b. The set of feasible directions in which to increase can be interpreted as the convex cone of feasible flows of an auxiliary network, that we introduce in the next section. We will thus be able to replace the hive inequalities (1) locally by capacity constraints.

5 The residual network

We start with a general definition. Given a polyhedron P in a real vector space V and a vector $f \in P$. We define the *cone of feasible directions* $C_f(P)$ *of* P *at* f as

$$C_f(P) := \{d \in V \mid \exists \varepsilon > 0 : f + \varepsilon d \in P\}.$$

We note that $P \cap U = (f + C_f(P)) \cap U$ for a small neighborhood U of f.

Recall that a rhombus $\Diamond$ is called *f-flat* with respect to a flow f iff $\sigma\big(\Diamond, f\big) = 0$. The flow f is called *shattered* iff there is no pair of overlapping f-flat rhombi.

Lemma 8 (Shattering) *Let λ, μ, ν be strictly decreasing partitions and let $f \in P^b$. Then one can algorithmically find a shattered integral flow* $\mathrm{shatter}(f) \in P^b$ *such that* $\delta(\mathrm{shatter}(f)) = \delta(f)$.

Proof: This was basically shown by Buch in his proof of the Saturation Property (Buc00). □

Fix a b-bounded shattered hive flow f. We now introduce the residual network $\mathsf{RES}^b(f)$.

The *residual digraph* $\mathsf{RES}(f)$ w.r.t. f is constructed as follows. The vertex and edge set of $\mathsf{RES}(f)$ are initially the vertex and edge set of G. Then each f-flat rhombus $\Diamond = \Diamond(\underline{A}, \underline{B}, \underline{C}, \underline{D})$ is replaced by the digraph illustrated in Figure 4. We remove all inner vertices of $\Diamond$, keep $[\underline{A}, \underline{B}]$, $[\underline{C}, \underline{B}]$, $[\underline{D}, \underline{C}]$ and $[\underline{D}, \underline{A}]$,

and add 14 vertices as in the figure. Then we add edges, some of which are fat black, some of which are fat white and some of which are normal as in the figure.

Note that in $\mathsf{RES}(f)$, the circle vertex $[\underline{B}, \underline{D}]$ is no longer present. The graph $\mathsf{RES}(f)$ is still bipartite, but may not be planar.

We next define the *residual network* $\mathsf{RES}^b(f)$. For each fat black edge e we set $l_e := 0$. This enforces that a well-directed cycle can only pass such e in the direction of e (compare Definition 4). For each fat white edge e we set $u_e := 0$. This enforces that a well-directed cycle can only pass such e in the reverse direction of e.

We now introduce additional capacities that are dependent on b. For the edges $e = \{o, s\}$, $s \in \mathscr{S}$, that are used to capacity with respect to the flow f in G^b we set $u_e := 0$. Moreover, for the edges $e = \{o, t\}$, $t \in \mathscr{T}$, that are used to capacity with respect to the flow f in G^b we set $l_e := 0$. We note that the feasible flows on $\mathsf{RES}^b(f)$ form a convex cone.

The following lemma shows that feasible flows on $\mathsf{RES}^b(f)$ give the directions from $f \in P^b$ that do not point out of P^b.

Lemma 9 (Residual Correspondence) *Let f be a b-bounded shattered hive flow. There is a natural surjective linear map*

$$\gamma\colon F(\mathsf{RES}(f)) \to F(G)$$

that preserves the throughput on all circle vertices that are both in $\mathsf{RES}(f)$ and G. We have

$$P_{\mathrm{feas}}(\mathsf{RES}^b(f)) = \gamma^{-1}(C_f(P^b)).$$

The map γ preserves integrality and the global throughput δ. □

For example, the flow $\gamma(f)(e)$ on the edge e directed away from $[\underline{A}, \underline{B}]$ in G is just the sum of flows $f(e_1) + f(e_2) + f(e_3)$ of the edges e_i directed away from $[\underline{A}, \underline{B}]$ in $\mathsf{RES}(f)$.

The following lemma gives an optimality criterion for optimizing the linear function δ on P^b.

Lemma 10 (Optimality Test) *Let f be a shattered b-bounded hive flow and let $\delta\colon F(G) \to \mathbb{R}$ be a linear function. Then f maximizes δ on P^b iff $\mathsf{RES}^b(f)$ has no well-directed cycle c with $\delta(\gamma(c)) > 0$.*

Proof: As δ is linear, f does not maximize δ on P^b iff there exists $d \in F$ such that $d \in P^b - f$ and $\delta(d) > 0$. Since $P^b - f$ equals $C_f(P^b)$ in a small neighborhood of 0, the latter condition is equivalent to the existence of some $d \in C_f(P^b)$ with $\delta(d) > 0$. According to Lemma 9, this is equivalent to the existence of a some $d' \in P_{\mathrm{feas}}(\mathsf{RES}^b(f))$ with $\delta(\gamma(d')) > 0$. We now show that this is equivalent to the existence of a well-directed cycle c on $\mathsf{RES}^b(f)$ with $\delta(\gamma(c)) > 0$.

Let $d' \in P_{\mathrm{feas}}(\mathsf{RES}^b(f))$ with $\delta(\gamma(d')) > 0$. Lemma 5 states that d' can be decomposed as $d' = \sum_{i=1}^{M} \alpha_i c_i$ where c_i are well-directed cycles on $\mathsf{RES}^b(f)$ and $\alpha_i > 0$. Thus $\delta(\gamma(d')) = \sum_{i=1}^{M} \alpha_i \delta(\gamma(c_i))$ is positive and hence there is a well-directed cycle c_i with $\delta(\gamma(c_i)) > 0$.

Conversely, if c is a well-directed cycle on $\mathsf{RES}^b(f)$ with $\delta(\gamma(c)) > 0$, then εc is a feasible flow on $\mathsf{RES}^b(f)$ for sufficiently small $\varepsilon > 0$. □

6 The main algorithm LRPA

The algorithm LRPA (Littlewood-Richardson Positivity Algorithm) is listed below.

Algorithm LRPA

Input: $\lambda, \mu, \nu \in \mathbb{N}^n$ strictly decreasing partitions with $|\nu| = |\lambda| + |\mu|$.
Output: Decide whether $c^{\nu}_{\lambda\mu} > 0$.

1: Create the regular target border b and the digraph G.
2: Start with $f \leftarrow 0$, `done` $\leftarrow$ **false**.
3: **while** not `done` **do**
4: $f \leftarrow \mathrm{shatter}(f)$.
5: Construct $\mathsf{RES}^b(f)$.
6: **if** there is a well-directed cycle in $\mathsf{RES}^b(f)$ with $\delta(\gamma(c)) > 0$ **then**
7: Find a shortest well-directed cycle c in $\mathsf{RES}^b(f)$ with $\delta(\gamma(c)) > 0$.
8: Augment 1 unit over c: $f \leftarrow f + \gamma(c)$.
9: // We have $f \in P^b$, due to Theorem 11.
10: **else**
11: `done` $\leftarrow$ **true**.
12: **end if**
13: **end while**
14: **if** $\delta(f) = |\nu|$ **then return true**.
15: **else return false**.

The shattering in line 4 is done by the algorithm mentioned in Lemma 8. Searching for shortest well-directed cycles in line 7 with positive δ-value can be done by a variant of the well-known Bellman-Ford algorithm (CLRS01).

The most interesting property of LRPA is that *shortest* well-directed cycles on $\mathsf{RES}^b(f)$ can be used to increase $\delta(f)$ by *one unit* (see line 8) while still remaining in P^b. The reason for this is the following crucial Theorem 11.

Theorem 11 (Shortest Cycle) *Let f be a b-bounded integral shattered hive flow. Assume that c is a shortest cycle among all well-directed cycles $\tilde{c}$ on $\mathsf{RES}^b(f)$ with $\delta(\tilde{c}) > 0$. Then $f + \gamma(c) \in P^b$.*

Proof sketch: The proof is rather involved. Assume that c is a cycle on $\mathsf{RES}^b(f)$ with $\delta(c) > 0$ and $f + \gamma(c) \notin P^b$. Let $\varepsilon := \max\{\varepsilon' \in \mathbb{R} \mid f + \varepsilon'\gamma(c) \in P^b\}$ and put $g := f + \varepsilon\gamma(c)$. We can show that $\varepsilon \in \{\frac{1}{3}, \frac{1}{2}, \frac{2}{3}\}$. Rhombi which are not f-flat, but g-flat, are called *critical rhombi*. Since $f + \gamma(c) \notin P^b$, there is at least one critical rhombus. A *g-flatspace* is a maximal connected union of small triangles such that any rhombus contained in it is g-flat.

In the case where there exists a critical rhombus that is not overlapping with any other g-flat rhombi, we can find well-directed cycles c_1, c_2 on $\mathsf{RES}^b(f)$ that split c in the sense that $\gamma(c_1 + c_2) = \gamma(c)$ and $\ell(c_1) \leq \ell(c_2) < \ell(c)$. One of these cycles must satisfy $\delta(c_i) > 0$ and we are done, because we found a cycle with positive throughput that is shorter than c.

In the other case we have overlapping g-flat rhombi and thus g is not shattered. So we can find a chain of g-flatspaces as described in (Buc00). We get a flow Ψ on G corresponding to raising the inner vertices

of the chain by one unit. Moreover $g + \varepsilon\Psi \in P^b$ and $\delta(\Psi) = 0$. Our goal is to find well-directed cycles $c_1, \ldots, c_m$ such that $\gamma(\sum_i c_i) = \gamma(c) + \Psi$ with $\ell(c_i) < \ell(c)$ for all i. Then we are done, because one of these cycles must satisfy $\delta(c_i) > 0$.

We achieve this by a complete classification of the possible shapes of g-flatspaces and the possible ways in which c passes through those shapes. Having understood this, we can do local changes to c such that it decomposes into smaller cycles that have the desired property. □

Theorem 11 allows us to prove the correctness of LRPA.

Theorem 12 *If given as input three strictly decreasing partitions $\lambda, \mu, \nu \in \mathbb{N}^n$ with $|\nu| = |\lambda| + |\mu|$, then the LRPA returns true iff $c^\nu_{\lambda\mu} > 0$.*

Proof: Note that during the algorithm f stays integral all the time and $f \in P^b$, because shattering preserves these properties according to Lemma 8 and we have $f + \gamma(c) \in P^b$ according to Theorem 11. After the while loop, the flow f maximizes δ on P^b according to Lemma 10. Lemma 7 tells us that $c^\nu_{\lambda\mu} > 0$ iff $\delta(f) = |\nu|$. □

7 Polynomial running time

We briefly sketch the ideas needed to transform LRPA into a combinatorial polynomial-time algorithm.

Theorem 13 *There is a polynomial-time algorithm that decides for given partitions $\lambda, \mu, \nu \in \mathbb{N}^n$ with $|\nu| = |\lambda| + |\mu|$ whether $c^\nu_{\lambda\mu} > 0$. The running time is polynomial in n and $\log|\nu|$.*

Proof sketch: We first note that by a perturbation argument, the general case can be reduced to the case where all the input partitions λ, μ, ν are strictly decreasing. This is done by exhibiting a special target border $\overline{b}$ such that for any partitions $\lambda, \mu, \nu \in \mathbb{N}^n$ and N sufficiently large, but of bitsize polynomial in n, we have $c^\nu_{\lambda\mu} > 0$ iff $c^{\tilde\nu}_{\tilde\lambda\tilde\mu} > 0$, where $\tilde\lambda = N\lambda + \overline{\lambda}$, $\tilde\mu = N\mu + \overline{\mu}$, and $\tilde\nu = N\nu + \overline{\nu}$.

We use a scaling method similar to that described in (AMO93) in the scaling of the Ford-Fulkerson algorithm. For $z \in \mathbb{R}$ a flow f is called *z-integral* iff it only takes values that are integral multiples of z. The algorithm now works as follows:

Put $k \leftarrow \lceil \log|\nu| \rceil + 1$. We efficiently construct an initial 2^k-integral $f \in P^b$ that is shattered and has regular border.

(*) We construct a modification $\mathsf{RES}^b_{2^k}(f)$ of the residual network $\mathsf{RES}^b(f)$ that excludes certain circle border vertices. Then we search for a shortest well-directed cycle c in $\mathsf{RES}^b_{2^k}(f)$ with $\delta(\gamma(c)) > 0$. If there is no such cycle, we set $k \leftarrow k - 1$ and go to (*). Otherwise we augment 2^k units over c: $f \leftarrow f + 2^k\gamma(c)$. A variation of Theorem 11 guarantees that f is still in P^b. Moreover, by construction of $\mathsf{RES}^b_{2^k}(f)$, f has still regular border. By an auxiliary optimization procedure, we can turn f into a *shattered* 2^k-integral flow in P^b. This is a refinement of Lemma 8 for which we need regularity on the border, as long as $k > 0$. We decrease now $k \leftarrow k - 1$ and go to (*).

The algorithm terminates when $k < 0$. Its output is an integral flow $f \in P^b$ with optimal δ-value. We have $c^\nu_{\lambda\mu} > 0$ iff $\delta(f) = |\nu|$ by Lemma 7. The algorithm can be shown to work in polynomial time. □

8 Deciding multiplicity freeness

Let $\mathsf{RES}_\times(f)$ denote the network that results from deleting in $\mathsf{RES}^b(f)$ the vertex o and all incident edges. Note that $\mathsf{RES}_\times(f)$ is independent of b.

The proof of Theorem 11 also yields the following result.

Proposition 14 *Given a b-bounded integral shattered hive flow f and a shortest well-directed cycle c on $\mathsf{RES}_\times(f)$. Then $f + \gamma(c) \in P^b$.* □

Corollary 15 *Let f be a b-bounded integral shattered hive flow with $\delta(f) = |\nu|$. Then we have $c^\nu_{\lambda\mu} > 1$ if and only if there exists a well-directed cycle in $\mathsf{RES}_\times(f)$.*

Proof: Suppose that c is a shortest well-directed cycle in $\mathsf{RES}_\times(f)$. Proposition 14 tells us that $g := f + \gamma(c)$ lies in P^b. It is easy to see that $\gamma(c) \neq 0$ and $\delta(\gamma(c)) = 0$. Hence g is another integral flow on P^b with throughput $|\nu|$. Lemma 7 implies $c^\nu_{\lambda\mu} > 1$.

To show the converse, suppose that $c^\nu_{\lambda\mu} > 1$. Lemma 7 implies that there exists an integral flow $g \in P^b$, $g \neq f$, with $\delta(g) = |\nu|$. The flow $d := g - f$ satisfies $\delta(d) = 0$ and hence uses no circle border vertex, which means that its support lies inside Δ. By Lemma 9 there exists $d' \in P_{\mathrm{feas}}(\mathsf{RES}^b(f))$ such that $d = \gamma(d')$. It is obvious that in fact $d' \in P_{\mathrm{feas}}(\mathsf{RES}_\times(f))$. Decomposing d' according to Lemma 5 shows the existence of a well-directed cycle in $\mathsf{RES}_\times(f)$. □

For strictly decreasing partitions we get a new proof of Fulton's conjecture, first shown by Knutson, Tao and Woodward (KTW04).

Corollary 16 *Let λ, μ, ν be strictly decreasing partitions with $|\nu| = |\lambda| + |\mu|$. Then the following three conditions are equivalent:*

$$(1)\ c^\nu_{\lambda\mu} = 1, \quad (2)\ \exists N\ c^{N\nu}_{N\lambda N\mu} = 1, \quad (3)\ \forall N\ c^{N\nu}_{N\lambda N\mu} = 1.$$

Proof: It suffices to show the implication from (2) to (3). Suppose that $c^{N\nu}_{N\lambda N\mu} = 1$ for some N, hence $c^\nu_{\lambda\mu} > 0$ by the saturation property. Since $c^\nu_{\lambda\mu} > 1$ implies $c^{N\nu}_{N\lambda N\mu} > 1$ we must have $c^\nu_{\lambda\mu} = 1$. Let f be a $b(\lambda,\mu,\nu)$-bounded integral shattered hive flow f with $\delta(f) = |\nu|$. Corollary 15 says that $\mathsf{RES}_\times(f)$ has no well-directed cycle. Since $\mathsf{RES}_\times(f) = \mathsf{RES}_\times(N'f)$ for all N', $\mathsf{RES}_\times(N'f)$ contains no well-directed cycle as well. Corollary 15 implies now that $c^{N'\nu}_{N'\lambda N'\mu} = 1$. □

Theorem 17 *There is a polynomial-time algorithm that decides for given strictly decreasing partitions $\lambda, \mu, \nu \in \mathbb{N}^n$ with $|\nu| = |\lambda| + |\mu|$ whether $c^\nu_{\lambda\mu} = 1$. The running time is polynomial in n and $\log|\nu|$.*

Proof: Using the algorithm of Theorem 13 one can compute an integral shattered $f \in P^b$ with $\delta(f) = |\nu|$. It is easy to check in polynomial time whether $\mathsf{RES}_\times(f)$ contains a well-directed cycle. Hence the assertion follows with Corollary 15. □

References

[AMO93] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows: theory, algorithms, and applications*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[Buc00] Anders Skovsted Buch. The saturation conjecture (after A. Knutson and T. Tao) with an appendix by William Fulton. *Enseign. Math.*, 2(46):43–60, 2000.

[CLRS01] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 2nd edition, 2001.

[CS84] Michael Clausen and Friedrich Stötzer. Pictures and Standardtableaux—Grundlagen und Algorithmen. *Bayreuth. Math. Schr.*, (16):1–122, 1984.

[Ful00] William Fulton. Eigenvalues, invariant factors, highest weights, and Schubert calculus. *Bull. Amer. Math. Soc. (N.S.)*, 37(3):209–249, 2000.

[GLS93] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*. Springer-Verlag, Berlin, 1993.

[Ike08] Christian Ikenmeyer. On the complexity of computing Kronecker coefficients and deciding positivity of Littlewood-Richardson coefficients. Master's thesis, Universität Paderborn, 2008. Online available at `http://math-www.uni-paderborn.de/agpb/work/ikenmeyer_diplom.pdf`.

[KT99] Allen Knutson and Terence Tao. The honeycomb model of $\mathrm{GL}_n(\mathbf{C})$ tensor products. I. Proof of the saturation conjecture. *J. Amer. Math. Soc.*, 12(4):1055–1090, 1999.

[KTT04] R. C. King, C. Tollu, and F. Toumazet. Stretched Littlewood-Richardson and Kostka coefficients. In *Symmetry in physics*, volume 34 of *CRM Proc. Lecture Notes*, pages 99–112. Amer. Math. Soc., Providence, RI, 2004.

[KTW04] Allen Knutson, Terence Tao, and Christopher Woodward. The honeycomb model of GL(n) tensor products II: Puzzles determine facets of the Littlewood-Richardson cone. *J. Amer. Math. Soc.*, 17(1):19–48, 2004.

[MS05] Ketan D. Mulmuley and Milind Sohoni. Geometric complexity theory III: On deciding positivity of Littlewood-Richardson coefficients. cs.ArXive preprint cs.CC/0501076, 2005.

[Nar06] Hariharan Narayanan. On the complexity of computing Kostka numbers and Littlewood-Richardson coefficients. *J. Algebraic Combin.*, 24(3):347–354, 2006.

[PV05] Igor Pak and Ernesto Vallejo. Combinatorics and geometry of Littlewood-Richardson cones. *Eur. J. Comb.*, 26(6):995–1008, 2005.

[Sta99] Richard P. Stanley. *Enumerative Combinatorics Volume 2*, volume 62 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, 1999.