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

Peter Bürgisser and Christian Ikenmeyer

UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

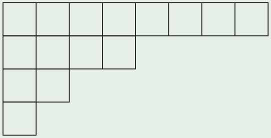
MFQI Sevilla, Nov. 2009

Definition

Given partitions λ, μ, ν, then the sequence of Kronecker coefficients $(k_{\lambda(n),\mu(n),\nu(n)})$ stabilizes, where $\lambda(n) := (n - |\lambda|, \lambda)$ denotes the partition of n that equals λ with additional first row.

Example

$\lambda = (4,2,1)$, $n = 15$. Then $\lambda(n)$ has the following Young diagram:

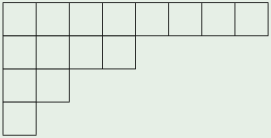


Definition

Given partitions λ, μ, ν, then the sequence of Kronecker coefficients $(k_{\lambda(n),\mu(n),\nu(n)})$ stabilizes, where $\lambda(n) := (n - |\lambda|, \lambda)$ denotes the partition of n that equals λ with additional first row.

Example

$\lambda = (4, 2, 1)$, $n = 15$. Then $\lambda(n)$ has the following Young diagram:



- Brion 1993, Vallejo 1999 and Briand, Orellana and Rosas 2009 gave upper bounds for n from which on the sequence is stable.

Definition

Given partitions λ, μ, ν with $|\nu| = |\lambda| + |\mu|$, then $\left(k_{\lambda(n),\mu(n),\nu(n))}\right)$ stabilizes to the **Littlewood-Richardson coefficient** $c^{\nu}_{\lambda\mu}$.

Definition

Given partitions λ, μ, ν with $|\nu| = |\lambda| + |\mu|$, then $\left(k_{\lambda(n),\mu(n),\nu(n))}\right)$ stabilizes to the **Littlewood-Richardson coefficient** $c^{\nu}_{\lambda\mu}$.

- Wide variety of interpretations in combinatorics, representation theory, geometry and in the theory of symmetric functions.

Definition

Given partitions λ, μ, ν with $|\nu| = |\lambda| + |\mu|$, then $\left(k_{\lambda(n),\mu(n),\nu(n)}\right)$ stabilizes to the **Littlewood-Richardson coefficient** $c^{\nu}_{\lambda\mu}$.

- Wide variety of interpretations in combinatorics, representation theory, geometry and in the theory of symmetric functions.
- No polynomial-time algorithm for the computation of $c^{\nu}_{\lambda\mu}$ unless $\mathbf{P} = \mathbf{NP}$ (Narayanan 2006).

Definition

Given partitions λ, μ, ν with $|\nu| = |\lambda| + |\mu|$, then $(k_{\lambda(n),\mu(n),\nu(n)})$ stabilizes to the **Littlewood-Richardson coefficient** $c^{\nu}_{\lambda\mu}$.

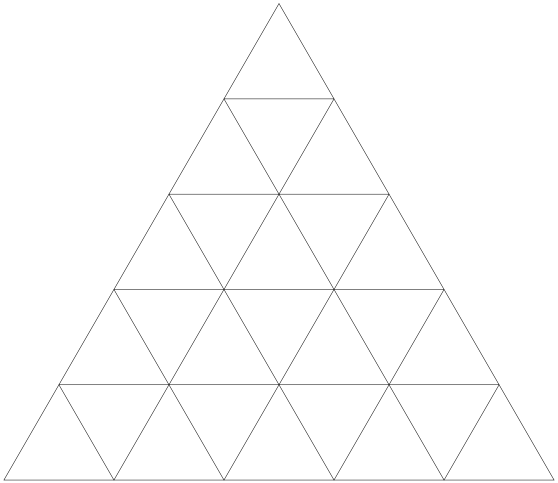
- Wide variety of interpretations in combinatorics, representation theory, geometry and in the theory of symmetric functions.
- No polynomial-time algorithm for the computation of $c^{\nu}_{\lambda\mu}$ unless $\mathbf{P} = \mathbf{NP}$ (Narayanan 2006).
- Problem $\mathrm{LR}_{>t}$:
  "For a given integer t, do we have $c^{\nu}_{\lambda\mu} > t$?".

- Problem $\mathrm{LR}_{>t}$:
  "For a given integer t, do we have $c^{\nu}_{\lambda\mu} > t$?".
- Knutson and Tao 1999, Mulmuley and Sohoni 2005: **$\mathrm{LR}_{>0}$ can be decided in polynomial time.**

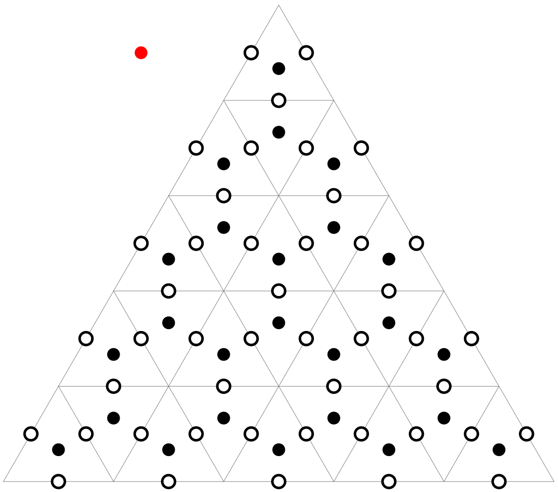
- Problem $\mathrm{LR}_{>t}$:
  "For a given integer t, do we have $c^{\nu}_{\lambda\mu} > t$?".
- Knutson and Tao 1999, Mulmuley and Sohoni 2005: **$\mathrm{LR}_{>0}$ can be decided in polynomial time.**
- Geometric Complexity Theory: Mulmuley and Sohoni in 2005 asked for a **combinatorial** polynomial-time algorithm for $\mathrm{LR}_{>0}$, like for max-flow or weighted matching problems in combinatorial optimization.

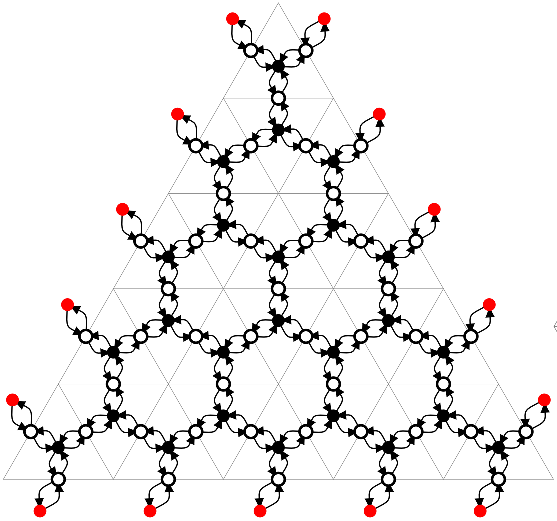
- Problem $\mathrm{LR}_{>t}$:
  "For a given integer t, do we have $c^{\nu}_{\lambda\mu} > t$?".
- Knutson and Tao 1999, Mulmuley and Sohoni 2005: **$\mathrm{LR}_{>0}$ can be decided in polynomial time.**
- Geometric Complexity Theory: Mulmuley and Sohoni in 2005 asked for a **combinatorial** polynomial-time algorithm for $\mathrm{LR}_{>0}$, like for max-flow or weighted matching problems in combinatorial optimization.
- Our contribution: A polynomial-time max-flow-type algorithm for $\mathrm{LR}_{>0}$ like requested by Mulmuley and Sohoni in 2005.

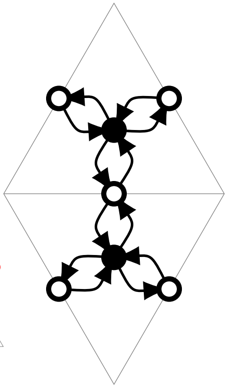
- Problem $\mathrm{LR}_{>t}$:
  "For a given integer t, do we have $c^{\nu}_{\lambda\mu} > t$?".
- Knutson and Tao 1999, Mulmuley and Sohoni 2005: **$\mathrm{LR}_{>0}$ can be decided in polynomial time.**
- Geometric Complexity Theory: Mulmuley and Sohoni in 2005 asked for a **combinatorial** polynomial-time algorithm for $\mathrm{LR}_{>0}$, like for max-flow or weighted matching problems in combinatorial optimization.
- Our contribution: A polynomial-time max-flow-type algorithm for $\mathrm{LR}_{>0}$ like requested by Mulmuley and Sohoni in 2005.
- Furthermore we developed an algorithm to decide $\mathrm{LR}_{>t}$ in time $\mathcal{O}(t^2 \mathrm{poly}(n))$.



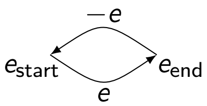
The graph Δ.







The digraph G.

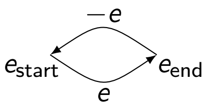


Definition (Flow)

A real mapping $f : E(G) \to \mathbb{R}$ satisfies the **flow constraints**, if for all vertices $v \in V(G)$ we have

$$\sum_{\substack{e \in E(G) \\ e_{\text{end}} = v}} f(e) = \sum_{\substack{e \in E(G) \\ e_{\text{start}} = v}} f(e).$$

These constraints define a subspace $U(G) \subset \mathbb{R}^{E(G)}$.



Definition (Flow)

A real mapping $f : E(G) \to \mathbb{R}$ satisfies the **flow constraints**, if for all vertices $v \in V(G)$ we have

$$\sum_{\substack{e \in E(G) \\ e_{\text{end}} = v}} f(e) = \sum_{\substack{e \in E(G) \\ e_{\text{start}} = v}} f(e).$$

These constraints define a subspace $U(G) \subset \mathbb{R}^{E(G)}$. Define the subspace

$$N(G) := \{f \in \mathbb{R}^{E(G)} \mid \forall e \in E(G) : f(e) = f(-e)\} \subset U(G)$$

generated by the 2-cycles. We set $\tilde{F}(G) := U(G)/N(G)$.

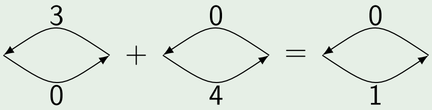
Definition (Flow)

Note that each coset of $\tilde{F}(G)$ contains **exactly one** element f that has

- only nonnegative flow values
- and $f(e) = 0$ or $f(-e) = 0$ for all edges e.

We call this system of representatives the **vector space $F(G)$ of flows on G**.

Example



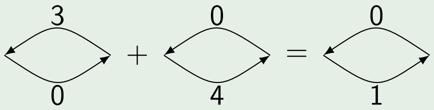
Definition (Flow)

Note that each coset of $\tilde{F}(G)$ contains **exactly one** element f that has

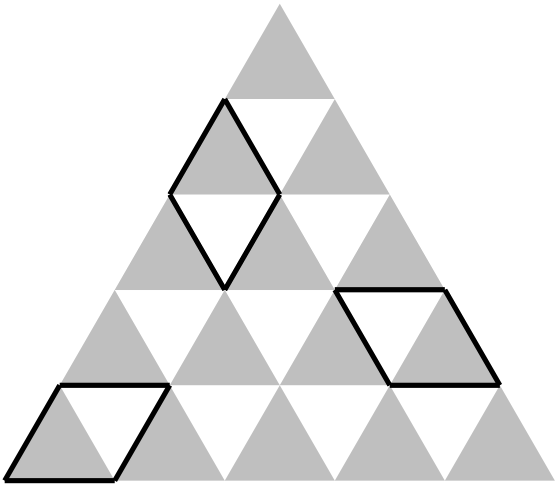
- only nonnegative flow values
- and $f(e) = 0$ or $f(-e) = 0$ for all edges e.

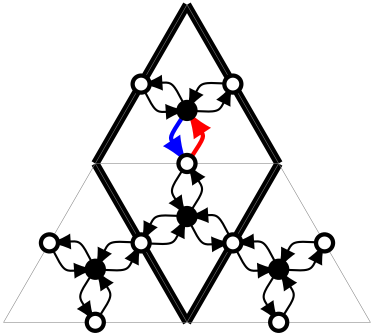
We call this system of representatives the **vector space $F(G)$ of flows on G**.

Example



- Canonical injection: (Oriented) cycles $C(G) \to$ Flows $F(G)$ (flow value of 1 on all cycle edges)





Define the throughput $\Diamond$ w.r.t. a flow $f \in F(G)$ as

$$\Diamond(f) := f(\textbf{blue}) - f(\textbf{red}).$$

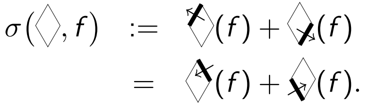
Analogously define $\Diamond$, $\Diamond$ and so on.

Define the **slack** σ of a rhombus $\Diamond$ w.r.t. a flow f as

$\sigma(\Diamond, f) :=$ $(f) +$ (f)

$=$ $(f) +$ $(f).$

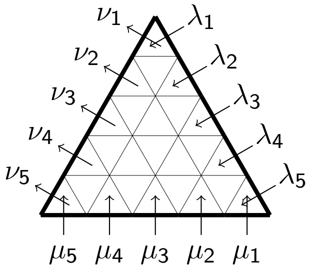
Define the **slack** σ of a rhombus $\lozenge$ w.r.t. a flow f as



$$\begin{aligned}\sigma(\lozenge, f) &:= \lozenge(f) + \lozenge(f)\\ &= \lozenge(f) + \lozenge(f).\end{aligned}$$

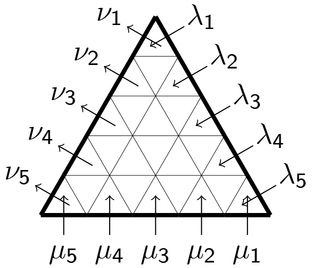
Definition (Hive flow)

We call a flow f a **hive flow**, if its slack w.r.t. all rhombi is nonnegative.



Theorem (Hive flow description)

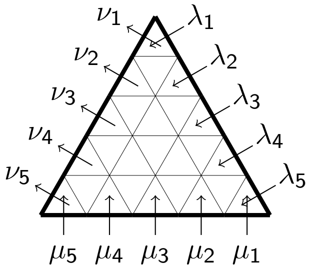
Given three partitions λ, μ and ν with $|\nu| = |\lambda| + |\mu|$, then the Littlewood-Richardson coefficient $c^{\nu}_{\lambda\mu}$ equals the number of **integral** *hive flows f with throughputs as in the figure.*



Theorem (Hive flow description)

Given three partitions λ, μ and ν with $|\nu| = |\lambda| + |\mu|$, then the Littlewood-Richardson coefficient $c^{\nu}_{\lambda\mu}$ equals the number of **integral** *hive flows f with throughputs as in the figure.*

Proof: Integral hive flows $\xleftrightarrow{\text{bij.}}$ integral hives by Knutson & Tao, Buch. $\square$

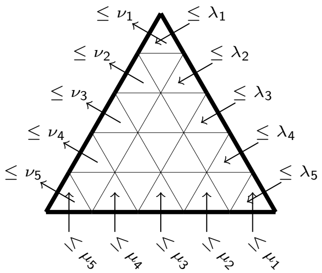


Theorem (Hive flow description)

Given three partitions λ, μ and ν with $|\nu| = |\lambda| + |\mu|$, then the Littlewood-Richardson coefficient $c^{\nu}_{\lambda\mu}$ equals the number of **integral** *hive flows f with throughputs as in the figure.*

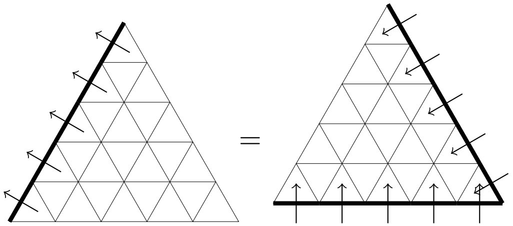
Proof: Integral hive flows $\overset{\text{bij.}}{\longleftrightarrow}$ integral hives by Knutson & Tao, Buch. $\square$

Flow description suitable for optimization techniques!



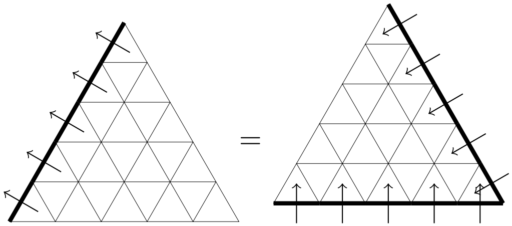
Definition (b-bounded hive flow)

Given a vector of three partitions $b{=}(\lambda, \mu, \nu)$ with $|\nu| = |\lambda| + |\mu|$, then a hive flow f is called ***b*-bounded**, if its throughputs satisfy the constraints in the figure.
P^b denotes the polyhedron of all b-bounded hive flows.



Definition (Overall throughput)

For a flow f on G we define $\delta(f)$ as the sum of throughputs in the figure.



Definition (Overall throughput)

For a flow f on G we define $\delta(f)$ as the sum of throughputs in the figure.

Lemma

1. *For all $f \in P^b$ we have $\delta(f) \leq |\nu|$.*
2. *$c_{\lambda\mu}^{\nu}$ equals the number of* **integral** *$f \in P^b$ with $\delta(f) = |\nu|$.*

Lemma

1. *For all $f \in P^b$ we have $\delta(f) \leq |\nu|$.*
2. *$c^{\nu}_{\lambda\mu}$ equals the number of* **integral** *$f \in P^b$ with $\delta(f) = |\nu|$.*

Algorithmic idea

```
$f \leftarrow 0$.
while $f$ is not maximal w.r.t. $\delta$ in $P^b$ do
    adjust $f \in P^b$ such that $f$ stays integral and in $P^b$ and $\delta(f)$
    increases by at least a fixed amount.
end while
We have that $f$ is maximal w.r.t. $\delta$ in $P^b$ and integral.
return whether $\delta(f) = |\nu|$.
```

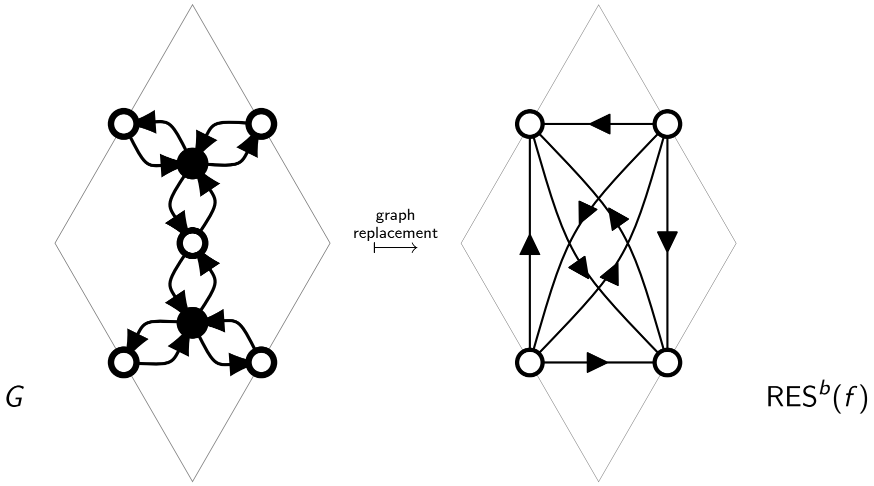
Lemma

For a given integral flow $f \in P^b$ one can algorithmically find an integral flow $g \in P^b$ with the same throughput and with **no overlapping rhombi that have zero slack***.*

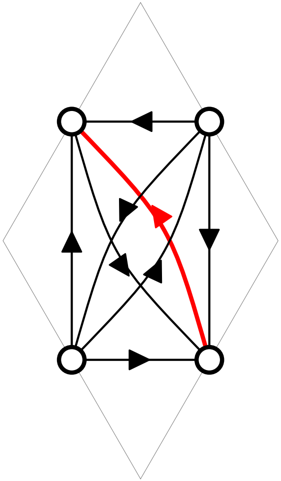
Proof mainly according to A. S. Buch 2000.

So assume for this talk that rhombi with zero slack **do not overlap**.

Replace each rhombus that has zero slack with the following graph:

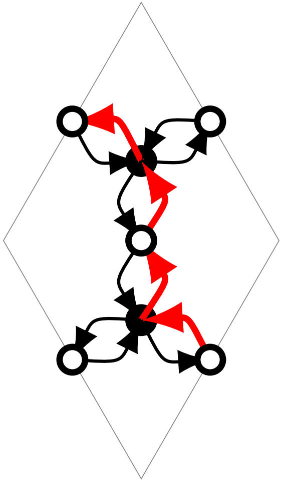


A flow on $\mathrm{RES}^b(f)$ induces a flow on G via a canonical map γ, which preserves the thoughputs on all vertices:



$\mathrm{RES}^b(f)$

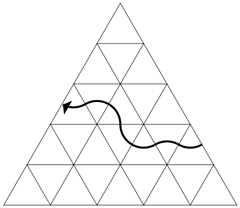
$\overset{\gamma}{\longmapsto}$

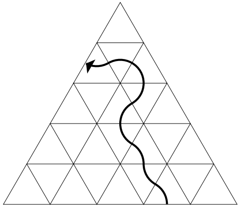


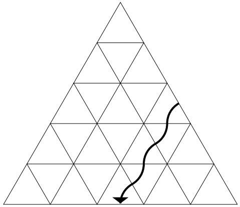
G

When does $\delta(f)$ increase by adding $\gamma(c)$?

$\delta\big(f+\gamma(c)\big) > \delta(f) \iff \delta\big(\gamma(c)\big) > 0 \iff\!:$ c is **δ-positive**

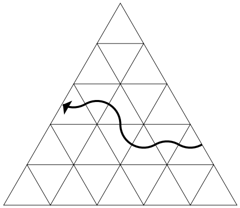

δ-positive


δ-positive

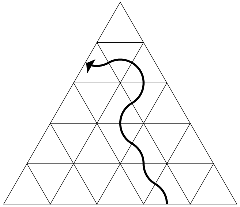

NOT δ-positive

When does $\delta(f)$ increase by adding $\gamma(c)$?

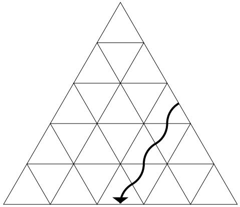
$\delta\big(f+\gamma(c)\big) > \delta(f) \iff \delta\big(\gamma(c)\big) > 0 \iff: c$ is **δ-positive**



δ-positive



δ-positive



NOT δ-positive

Theorem (Shortest Cycle Theorem)

Given an integral flow $f \in P^b$ and a δ-positive cycle c on $\mathrm{RES}^b(f)$, **shortest** *among all δ-positive cycles on $\mathrm{RES}^b(f)$, then $f+\gamma(c) \in P^b$.*

Theorem (Shortest Cycle Theorem)

Given an integral flow $f \in P^b$ and a δ-positive cycle c on $\mathrm{RES}^b(f)$, shortest among all δ-positive cycles on $\mathrm{RES}^b(f)$, then $f + \gamma(c) \in P^b$.

Algorithm LRPA

```
$f \leftarrow 0$.
**while** there is a $\delta$-positive cycle on $\mathrm{RES}^b(f)$ **do**
    search for a shortest $\delta$-positive cycle $c$ on $\mathrm{RES}^b(f)$.
    $f \leftarrow f + \gamma(c)$.
**end while**
**return** whether $\delta(f) = |\nu|$.
```

Theorem (Shortest Cycle Theorem)

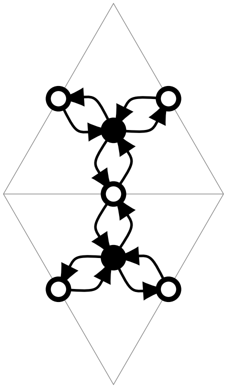
Given an integral flow $f \in P^b$ and a δ-positive cycle c on $\mathrm{RES}^b(f)$, shortest among all δ-positive cycles on $\mathrm{RES}^b(f)$, then $f + \gamma(c) \in P^b$.

Algorithm LRPA

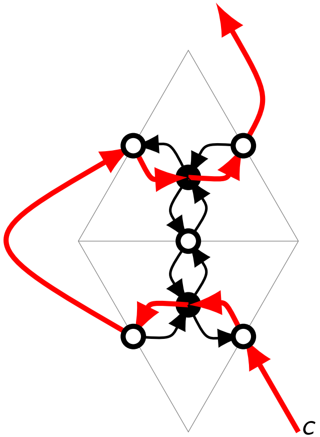
```
$f \leftarrow 0$.
**while** there is a $\delta$-positive cycle on $\mathrm{RES}^b(f)$ **do**
    search for a shortest $\delta$-positive cycle $c$ on $\mathrm{RES}^b(f)$.
    $f \leftarrow f + \gamma(c)$.
**end while**
**return** whether $\delta(f) = |\nu|$.
```

Lemma (Optimality Test)

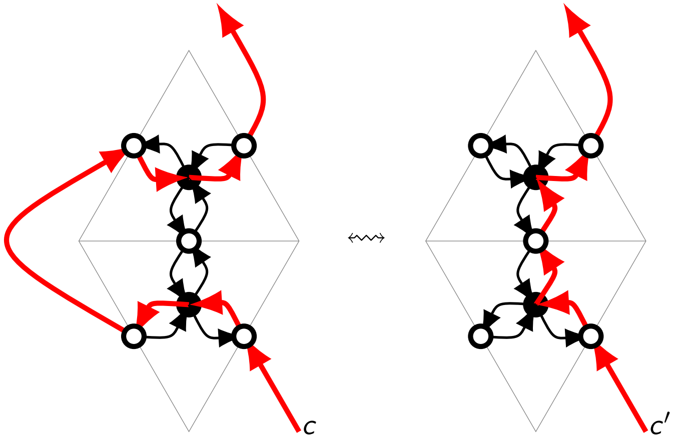
Given a flow $f \in P^b$, then f maximizes δ in P^b iff on $\mathrm{RES}^b(f)$ there is no δ-positive cycle.



Assume that there is no rhombus with zero slack and thus no subgraph replacement.
Let $\Diamond$ have slack $\sigma(\Diamond, f) = 1$.

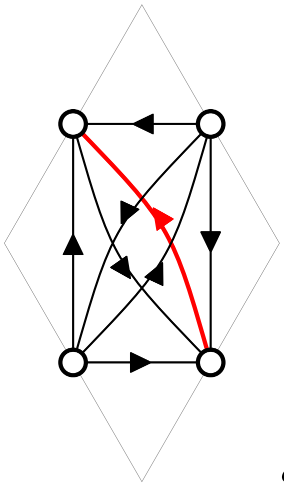


$\sigma(\Diamond, f) = 1$. Recall $\sigma(\Diamond, c) = \overset{\nwarrow}{\cancel{\Diamond}}(c) + \underset{\searrow}{\cancel{\Diamond}}(c) = -2$.
Hence $\sigma(\Diamond, f + c) = \sigma(\Diamond, f) + \sigma(\Diamond, c) = -1 < 0$
and thus $f + c$ is **not a hive flow**.



$f + c$ is not a hive flow, but c was not a **shortest** cycle.
$\sigma\left(\Diamond, c'\right) = \cancel{\Diamond}(c') + \cancel{\Diamond}(c') = -1$ and $f + c'$ is a hive flow, because
$\sigma\left(\Diamond, f + c'\right) = 0$.

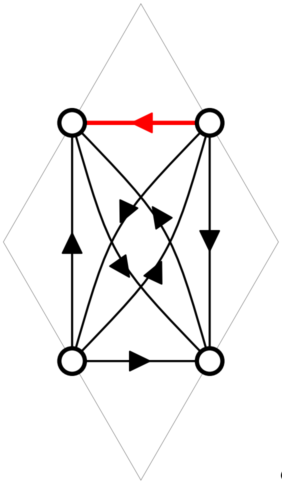
Now let $\sigma(\Diamond, f) = 0$ and thus the subgraph is replaced:



c on $\mathrm{RES}^b(f)$

$$\sigma(\Diamond, \gamma(c)) = \Diamond(c) + \Diamond(c) = 0.$$

Now let $\sigma(\Diamond, f) = 0$ and thus the subgraph is replaced:



c on $\mathrm{RES}^b(f)$

$$\sigma(\Diamond, \gamma(c)) = \Diamond(c) + \Diamond(c) = 1.$$

Lemma

The graph replacement ensures that all rhombi with $\sigma\big(\Diamond, f\big) = 0$ *have* $\sigma\big(\Diamond, f + \gamma(c)\big) \geq 0$ *for all cycles* c *on* $\mathrm{RES}^b(f)$.

Lemma

The graph replacement ensures that all rhombi with $\sigma\big(\Diamond, f\big) = 0$ *have* $\sigma\big(\Diamond, f + \gamma(c)\big) \geq 0$ *for all cycles* c *on* $\mathrm{RES}^b(f)$.

- There are more involved cases.
- Other problems arise when we have overlapping rhombi with zero slack.

Algorithm LRPA

```
$f \leftarrow 0$.
**while** there is a $\delta$-positive cycle on $\mathrm{RES}^b(f)$ **do**
    search for a shortest $\delta$-positive cycle $c$ on $\mathrm{RES}^b(f)$.
    $f \leftarrow f + \gamma(c)$.
**end while**
**return** whether $\delta(f) = |\nu|$.
```

Algorithm LRPA

```
$f \leftarrow 0$.
**while** there is a $\delta$-positive cycle on $\mathrm{RES}^b(f)$ **do**
  search for a shortest $\delta$-positive cycle $c$ on $\mathrm{RES}^b(f)$.
  $f \leftarrow f + \gamma(c)$.
**end while**
**return** whether $\delta(f) = |\nu|$.
```

Capacity scaling method (without technicalities)

```
$f \leftarrow 0$.
**for** $k$ down to 0 **do**
  **while** there is a $\delta$-positive cycle on $\mathrm{RES}^b_{2^k}(f)$ **do**
    search for a shortest $\delta$-positive cycle $c$ on $\mathrm{RES}^b_{2^k}(f)$.
    $f \leftarrow f + 2^k \cdot \gamma(c)$.
  **end while**
**end for**
**return** whether $\delta(f) = |\nu|$.
```

Theorem (Main Theorem)

The capacity scaling version of the LRPA decides $LR_{>0}$ in polynomial time.

For strictly decreasing partitions:

Corollary (Multiplicity freeness)

Let $f \in P^b$ integral with $\delta(f) = |\nu|$.
Then $c_{\lambda\mu}^{\nu} > 1$ iff there exists a cycle on $\mathrm{RES}^b(f)$.

For strictly decreasing partitions:

Corollary (Multiplicity freeness)

Let $f \in P^b$ integral with $\delta(f) = |\nu|$.
Then $c_{\lambda\mu}^{\nu} > 1$ iff there exists a cycle on $\mathrm{RES}^b(f)$.

Corollary

The capacity scaling version of the LRPA combined with the check for multiplicity freeness can decide whether $c_{\lambda\mu}^{\nu} = 0$, $c_{\lambda\mu}^{\nu} = 1$ or $c_{\lambda\mu}^{\nu} > 1$ in polynomial time.

For strictly decreasing partitions:

Corollary (Multiplicity freeness)

Let $f \in P^b$ integral with $\delta(f) = |\nu|$.
Then $c^{\nu}_{\lambda\mu} > 1$ iff there exists a cycle on $\mathrm{RES}^b(f)$.

Corollary (Fulton's Conjecture)

The following three conditions are equivalent:

1. $c^{\nu}_{\lambda\mu} = 1$,
2. $\exists N: \; c^{N\nu}_{N\lambda N\mu} = 1$,
3. $\forall N: \; c^{N\nu}_{N\lambda N\mu} = 1$.

First proved by Knutson, Tao and Woodward in 2004.

Not yet published:
We can define a more general residual network RES that allows to reach all δ-maximal flows in P^b by adding cycles in RES.
Efficient enumerating of these cycles results in:

Theorem

- *There exists an algorithm for deciding $LR_{>t}$ in time $\mathcal{O}(t^2 \mathrm{poly}(n))$.*
- *There exists an algorithm for computation of $c^{\nu}_{\lambda\mu}$ in time $\mathcal{O}\Big((c^{\nu}_{\lambda\mu})^2 \mathrm{poly}(n)\Big)$.*

Not yet published:
We can define a more general residual network RES that allows to reach all δ-maximal flows in P^b by adding cycles in RES.
Efficient enumerating of these cycles results in:

Theorem

- *There exists an algorithm for deciding $LR_{>t}$ in time $\mathcal{O}(t^2 \mathrm{poly}(n))$.*
- *There exists an algorithm for computation of $c^{\nu}_{\lambda\mu}$ in time $\mathcal{O}\Big((c^{\nu}_{\lambda\mu})^2 \mathrm{poly}(n)\Big)$.*

These algorithms efficiently enumerate all hive flows with maximal throughput for given λ, μ, ν.
They can also be used for efficient enumeration of all hive flows with maximal throughput for fixed λ, μ and variable ν.

Thank you.