Kostka Numbers and Littlewood-Richardson Coefficients: Distributed Computation

Janvier Nzeutchap
Laboratoire d'Informatique de l'Institut Gaspard-Monge (IGM), UMR-CNRS 8049
F-77454 Marne-la-Vallée cedex 2. FRANCE
jnzeutch@etudiant.univ-mlv.fr

Frédéric Toumazet & Franck Butelle
Laboratoire d'Informatique Paris Nord (LIPN), UMR-CNRS 7030
Avenue J.B. Clément, 93430 Villetaneuse. FRANCE
Frederic.Toumazet@lipn.univ-paris13.fr & Franck.Butelle@lipn.univ-paris13.fr

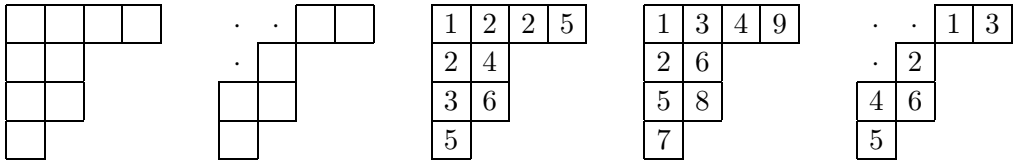
Abstract. Computing Kostka numbers and Littlewood-Richardson coefficients remains of great interest in combinatorics, and Brian G. Wybourne was among the first people to design software `-SCHUR-` for their computation. The efficiency of existing software `-SCHUR`, Stembridge package, `LattE`, Cochet's programs- is generally constrained by the lengths or weights of partitions. This work describes another method, based on the hives model, applying distributed computing techniques to the determination of generating polynomials for stretched Kostka numbers and stretched Littlewood-Richardson coefficients. This method can be used to "quickly" find such polynomials, with the help (of a predefined subset) of the available computers of the Local Area Network.

1. Definitions

A **partition** of a positive integer n is a way of writing n as a sum of non-increasing integers. For example $\lambda = (4, 2, 2, 1)$ and $\mu = (2, 1)$ are partitions of $n = 9$ and $n' = 3$ respectively. We write $\lambda \vdash n$ and $\mu \vdash n'$ or $|\lambda| = n$ and $|\mu| = n'$.

The **Ferrers diagram** F^λ associated to a partition $\lambda = (\lambda_1, \lambda_2, ..., \lambda_p)$ consists of $|\lambda| = n$ boxes, arranged in $\ell(\lambda) = p$ left-justified rows of lengths λ_1, λ_2, ..., λ_p. Rows in F^λ can be oriented downwards or upwards. F^λ is called the shape of λ. If F^λ contains F^μ, then the **skew diagram** (or skew partition) λ/μ is the one obtained from F^λ by deleting F^μ.

A **semi-standard Young tableau** of shape λ ($SSYT^\lambda$) is a numbering of the boxes of F^λ with entries from $\{1, 2, ..., n\}$, weakly increasing across rows and strictly increasing up (or down) columns. A tableau is **standard** (SYT^λ) if all its entries are different. Skew tableaux are defined in an analogous way. For example, for $\lambda = (4, 2, 2, 1)$ and $\mu = (2, 1)$, the Ferrers diagram F^λ, the skew Ferrers diagram λ/μ, a semi-standard tableau and a standard tableau both of shape λ, and a standard skew tableau of shape λ/μ are as follows:



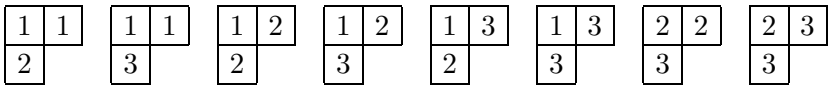
A **Symmetric Function** is a function which is symmetric or invariant under permutation of its variables.

$$f(x_{\sigma(1)}, x_{\sigma(2)}, ..., x_{\sigma(n)}) = f(x_1, x_2, ..., x_n) \tag{1}$$

where σ is any permutation of the symmetric group $\mathfrak{S}_\mathfrak{n}$. The **Schur function** s_λ is the symmetric function defined as:

$$s_\lambda(\mathsf{x}) = \sum_{SSYT^{\,\lambda}} x_1^{m_1}\, x_2^{m_2} \,...\, x_n^{m_n} \tag{2}$$

where $|\lambda| = n$, m_i is the number of entries i in $SSYT^{\,\lambda}$ for $i = 1, 2, ..., n$. For example, for $\mu = (2, 1)$ there are 8 semi-standard tableaux of shape μ:



So the Schur function $s_\mu(\mathsf{x})$ is the symmetric function defined as:

$$s_{21}(\mathsf{x}) = x_1^2\, x_2 + x_1^2\, x_3 + x_1\, x_2^2 + x_1\, x_2\, x_3 + x_1\, x_2\, x_3 + x_1\, x_3^2 + x_2^2\, x_3 + x_2\, x_3^2$$

Littlewood-Richardson coefficients $c^{\,\nu}_{\lambda\mu}$ are defined as the structure constants for the multiplication in the basis of Schur functions. So if $\lambda \vdash n$ and $\mu \vdash m$:

$$s_\lambda\, s_\mu = \sum_{\nu\,\vdash\, n+m} c^{\,\nu}_{\lambda\,\mu}\, s_\nu \tag{3}$$

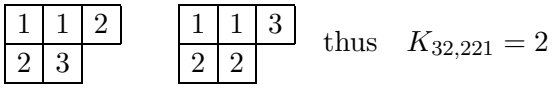
Example:
$s_{4221}\, s_{21} = s_{6321} + s_{6222} + s_{62211} + s_{5421} + s_{5331} + 2\, s_{5322} + 2\, s_{53211} + s_{4431} + s_{4422} + s_{44211} + 2\, s_{52221} + s_{4332} + 2\, s_{43221} + s_{522111} + s_{43311} + s_{432111} + s_{42222} + s_{422211}$

Thus $c^{\,5322}_{\,4221,\,21} = c^{\,43221}_{\,4221,\,21} = 2$ and $c^{\,6321}_{\,4221,\,21} = c^{\,422211}_{\,4221,\,21} = 1$.

Littlewood-Richardson coefficients are also defined as coefficients in the expansion of a skew Schur function in the basis of Schur functions:

$$s_{\nu/\lambda} = \sum_{\mu} c^{\,\nu}_{\lambda\,\mu}\, s_\mu \tag{4}$$

Kostka numbers $K_{\lambda\mu}$ are the number of distinctly labeled semi-standard Young tableaux of shape λ and weight μ, that is to say with μ_i entries i for $i = 1, 2, ..., \ell\,(\mu)$. For example, if $\lambda = (3, 2)$ and $\mu = (2, 2, 1)$ then there are 2 semi-standard Young tableaux ($SSYT^{\,32}$) of weight $(2, 2, 1)$:



thus $K_{32,221} = 2$

Kostka numbers can be used to give a combinatorial definition of Schur functions as linear combinations of monomial symmetric functions, as well as to express complete symmetric functions as combinations of Schur functions as follows [8]:

$$s_\lambda = \sum_{\mu\,\vdash\,|\lambda|} K_{\lambda\,\mu}\, m_\mu \quad \text{and} \quad h_\mu = \sum_{\lambda\,\vdash\,|\mu|} K_{\lambda\,\mu}\, s_\lambda \tag{5}$$

For each partition λ with $\ell(\lambda) \leq n$ there exists an irreducible representation V^λ of $GL(n)$ of highest weight λ and character ch_{V^λ} [9]. With a suitable identification of the indeterminate x_1, x_2, ..., x_n, the character ch_{V^λ} is nothing other than the Schur-function $s_\lambda(\mathsf{x})$. It follows that the Littlewood-Richardson coefficients govern the decomposition of tensor products of irreducible representations of $GL(n)$ in accordance with the formula:

$$V^\lambda \otimes V^\mu = \sum_\nu c^{\nu}_{\lambda\,\mu}\, V^{\,\nu} \tag{6}$$

Moreover [9], the weight space decomposition of the irreducible representation V^λ takes the form:

$$V^\lambda = \oplus\, \mathbf{m} V^{\,\lambda}_{\mathbf{m}} \tag{7}$$

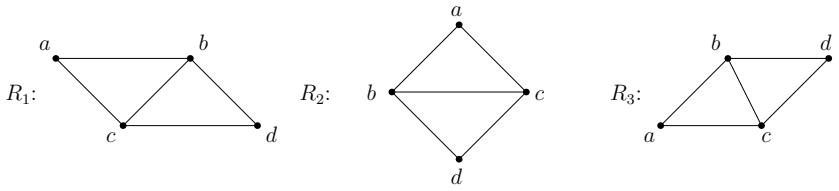
where the sum is taken over all weights $\mathbf{m} = (m_1, m_2, ..., m_n)$. This decomposition is such that the multiplicity of the weight specified by $\mathbf{m} = \mu$ in the irreducible representation V^λ is given by $\dim V^\lambda_\mu = K_{\lambda\mu}$.

2. Main results and motivations

Littlewood-Richardson coefficients $c^{\nu}_{\lambda\,\mu}$ have a polynomial growth with respect to the dilatation factor $N \in \mathbb{N}$:

$$c^{N\nu}_{N\lambda,\,N\mu} = P^{\,\nu}_{\lambda\,\mu}(N) \qquad ; \qquad P^{\,\nu}_{\lambda\,\mu}(0) = 1 \tag{8}$$

where $P^{\,\nu}_{\lambda\,\mu}$ is a polynomial in N with non negative rational coefficients depending on λ, μ and ν. This was first conjectured in [9]. A partial proof (existence of $P^{\,\nu}_{\lambda\,\mu}$ and rationality of coefficients) was given in [4]. Those polynomials [9] are obtained considering a model known as the hive model. An n**-integer-hive** is a triangular array of non negative integers a^i_j with $0 \leq i, j \leq n$ where neighboring entries define three distinct types of rhombus, each with its own constraint condition.



In each case, with the labeling as shown, the hive condition takes the form:

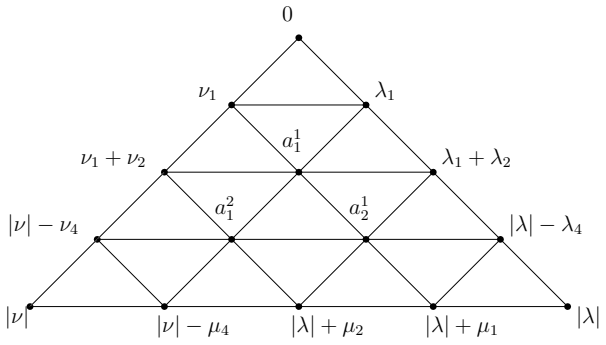
$$b + c \geq a + d \tag{9}$$

An ***LR*-hive** [9] is an integer hive satisfying the hive condition (9) for all its constituent rhombi of type R_1, R_2 and R_3, with border labels determined by the partitions λ, μ and ν, such that:

$$\begin{cases} a^0_0 = 0\ ;\ a^0_j = \lambda_1 + \lambda_2 + ... + \lambda_j & \text{for j = 1, 2, ..., } n \\ a^i_0 = \nu_1 + \nu_2 + ... + \nu_i & \text{for i = 1, 2, ..., } n \\ a^k_{n-k} = a^0_n + \mu_1 + \mu_2 + ... + \mu_k & \text{for k = 1, 2, ..., } n \end{cases}$$

with $\ell(\lambda)$, $\ell(\mu)$, $\ell(\nu) \leq n$ and $|\,\lambda\,| + |\,\mu\,| = |\,\nu\,|$.

For example, for $n = 4$, LR-hives will be sketched as:



The Littlewood-Richardson coefficient $c^{\nu}_{\lambda\,\mu}$ is the number of LR-hives with border labeled as above by λ, μ and ν. So $c^{\nu}_{\lambda\,\mu}$ is the number of triples (a_1^1, a_1^2, a_2^1) satisfying (9) for all rhombi of type R_1, R_2 and R_3. That is the number of integer solutions to the system of inequalities obtained by writing (9) for all rhombi. The degree of the polynomial $P^{\nu}_{\lambda\,\mu}(N)$ is always bounded by $(n-1)(n-2)/2$, the number of interior points of the hive, but if any of the partitions, λ, μ or ν has repeated parts then it is possible to use the hive inequalities to improve on this bound with some $maxDeg < (n-1)(n-2)/2$, see **Proposition 5.1** and **Conjecture 5.2** in [10]. Also see **Theorem 4.1** and **Corollary 4.2** in [4]. We can produce a *Maple* code to compute $c^{N\nu}_{N\lambda,\,N\mu}$ for any $N \in \mathbb{N}$. This code can be automatically translated into C and compiled. This increases the computation speed of individual coefficients, *Maple* being an interpreted language, contrary to C which is a compiled one. Although this code has been translated into C, the computation of a single coefficient still takes considerable time. Computing at most $maxDeg+1$ coefficients leads to the determination of P by interpolation. In fact only $maxDeg$ values are computed, since $P^{\nu}_{\lambda\,\mu}(0) = 1$.

Kostka numbers [9] can be viewed as special kinds of Littlewood-Richardson coefficients. To be precise, $K_{\lambda\,\mu} = c^{\tau}_{\sigma\,\lambda}$ where:

$$\begin{cases} \tau_i = \mu_i + \mu_{i+1} + ... & \text{for i = 1, 2, ..., } \ell(\mu) \\ \sigma_i = \mu_{i+1} + \mu_{i+2} + ... & \text{for i = 1, 2, ..., } \ell(\mu) - 1 \end{cases}$$

For example, if $\lambda = (3,2)$ and $\mu = (2,2,1)$, then $\tau = (5,3,1)$ and $\sigma = (3,1)$. Computing $c^{\tau}_{\sigma\,\lambda}$ by means of the Littlewood-Richardson rule leads to the diagrams:

It follows that $K_{32\,,221} = c^{531}_{61,\,32} = 2$.

A ***K*-hive** [9] is an integer hive satisfying the hive condition (9) for all its constituent rhombi of type R_1, R_2 but not necessarily R_3, with border labels determined by the partitions λ and μ, such that:

$$\begin{cases} a_0^i = 0 & \text{for i = 1, 2, ..., } n \\ a_j^0 = \lambda_1 + \lambda_2 + ... + \lambda_j & \text{for j = 1, 2, ..., } n \\ a_{n-k}^k = \mu_1 + \mu_2 + ... + \mu_k & \text{for k = 1, 2, ..., } n \end{cases}$$

with $\ell(\lambda)$, $\ell(\mu) \leq n$ and $|\lambda| = |\mu|$.

3. Main steps of the distributed computation

The main idea is to achieve the computation of the $maxDeg$ necessary coefficients with the help (of a predefined subset) of the available computers of the Local Area Network. We now describe step-by-step the distributed computation process.

step 1. The triple (λ, μ, ν) is given as an input to a first `Maple` program we've called `Hives`. `Hives` constructs the hive corresponding to λ, μ and ν, and extract from the hive a system of inequalities defining a convex polytope. It also produces a *Maple* code of a function to compute $c_{N\lambda,\, N\mu}^{N\nu}$ for any $N \in \mathbb{N}$, that is a function to count integer points in the dilated polytope. This program will be called `Prog1`. `Hives` also produces the value of $maxDeg$.
Of course how `Prog1` counts those points is a crucial question. This is a completely non-trivial problem on its own. The problem in question is in fact a well-known *P-hard* problem (in the computer science sense), but it is not the purpose of the paper to explore this. In practice, only the two matrices A and B defining the system of inequalities $AX \leq B$ corresponding to the primary polytope will be determined. So `Prog1` should be regarded as a generic program to solve linear systems of inequalities. We envisage passing those matrices to some well known such programs, in particular we've already considered formatting them as input files for `LattE` [3]. The difficulty here rests on the fact that all of the available software to compute these coefficients is efficient only in certain particular cases. This is the case in [2] where the computation times are reasonable only so far as the lengths of the partitions are less than or equal to 6. Another idea will be to use some other software to compute stretched Littlewood-Richardson coefficients for N from 1 to $maxDeg$. We plan to start this with `lrcalc` (Littlewood-Richardson Calculator) of A. S Buch [1].
On the occasion of this commemorative meeting organized for the memory of Brian G. Wybourne, we chose to use some programs he designed a few years ago, programs of which we maintain and upgrade a copy for experimental purpose. This iterative program rapidly becomes inefficient as $|\lambda|$, $|\mu|$ and $|\nu|$ grow.

step 2. `Prog1` is given as an input to a second `Maple` program which translates it into C as `Prog2`. So we are now provided with a couple (`Prog2`, $maxDeg$) where `Prog2` is the C code of a function to compute $c_{N\lambda,\, N\mu}^{N\nu}$ for any $N \in \mathbb{N}$, and $maxDeg$ is the max degree of the polynomial P. That is sufficient to distribute computations through the network:

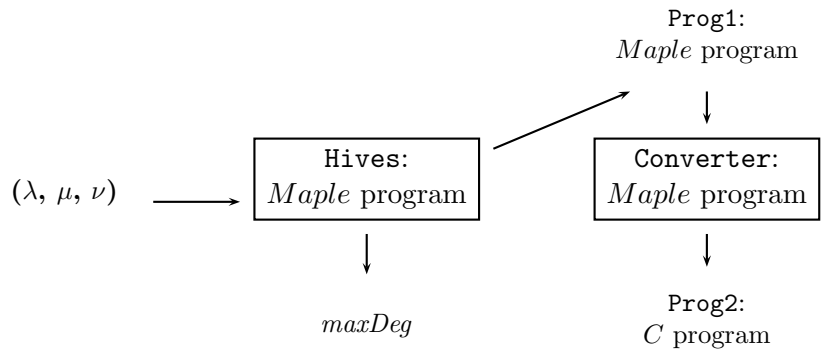


Fig. 1: automatic generation of code, determination of *maxDeg*.

step 3. `Prog2` is sent to a subset of the computers of the LAN, each of which is running a client application. Let us insist on the fact that `Prog2` can be any other program to compute Littlewood-Richardson coefficients.

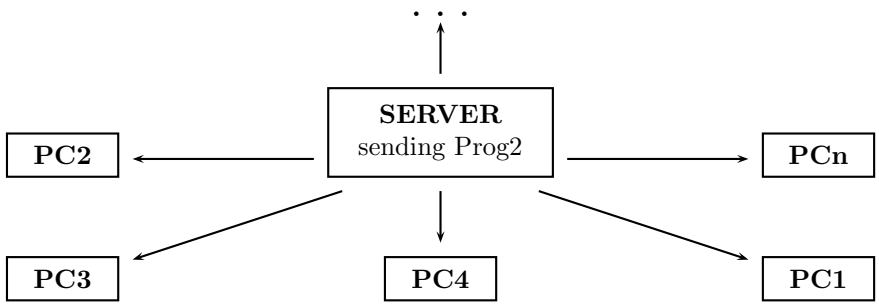


Fig. 2: the server sends the C code to computers on the LAN.

step 4. When a computer receives the C code (`Prog2`), it compiles it and sends an acknowledgement to the server to inform him of being ready for computations:

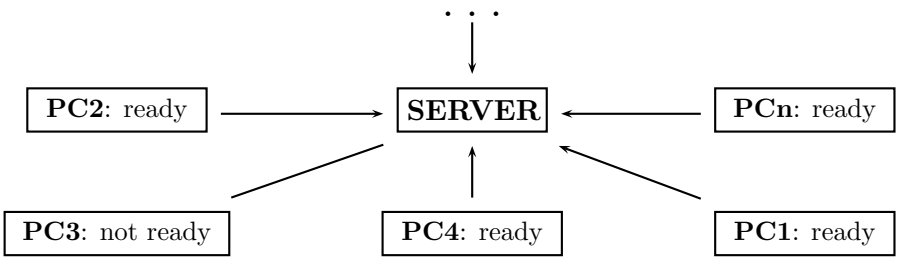


Fig. 3: the server receives acknowledgements from some computers.

Note that in the situation described on the figure above, the calculator named PC3 did not send an acknowledgement to the server.

step 5. Any computer which is ready receives a dilatation factor for which it is ordered to compute the corresponding coefficient. Note that PC3 has just sent an acknowledgement to the server. It has not yet received any dilatation factor, and will receive one as soon as the server takes note of its acknowledgement. The distributed computation is *asynchronous*.

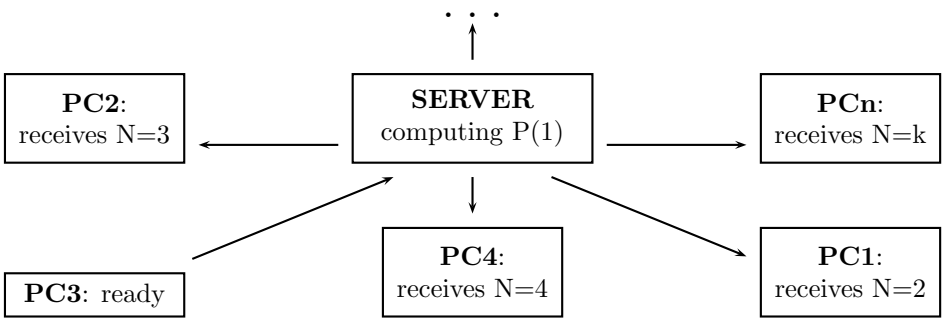


Fig. 4: the server sends dilatation factors to different computers.

step 6. When a computation is done, a computer sends its result to the server and receives the next dilatation factor for which it starts a new computation (Fig. 5 and Fig. 6).

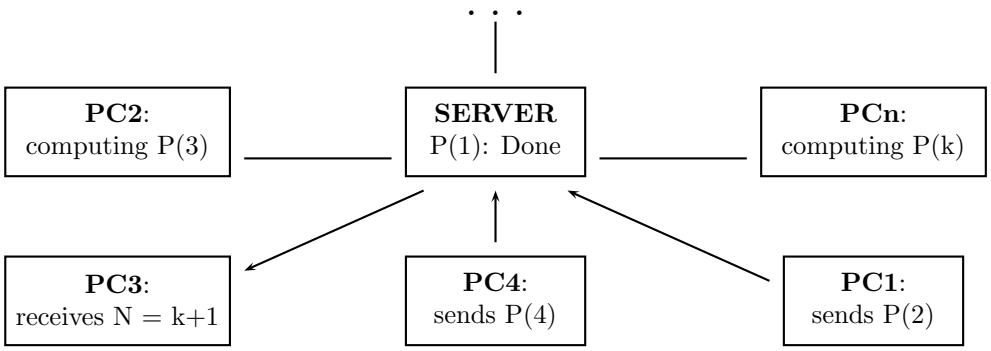


Fig. 5: results are sent back to the server.

Note that PC3 is receiving its first dilatation factor at the moment $PC1$ and $PC4$ are sending their first computation results to the server.

step 7. Each time the server receives new values, it makes an interpolation, using the list of all available values. So a set of polynomials

$$(P_i{}^{\nu}_{\lambda\mu}) = P_1{}^{\nu}_{\lambda\mu}\ ,\ P_2{}^{\nu}_{\lambda\mu}\ ,\ \ldots\ ,\ P_{maxDeg}{}^{\nu}_{\lambda\mu} \tag{10}$$

is constructed. **Experimentally, this set is often stationary**. This means that for a given triple (λ, μ, ν), there is an integer i_0 probably depending on λ, μ and ν, such that for any $i \geq i_0$ the equality $P_i{}^{\nu}_{\lambda\mu} = P_{i_0}{}^{\nu}_{\lambda\mu}$ holds. So we usually don't need to interpolate the theoretical maximum number of points to get the final polynomial. In other words, the polynomial $P^{\nu}_{\lambda\mu}$ usually has lower degree than the maximum degree expected.

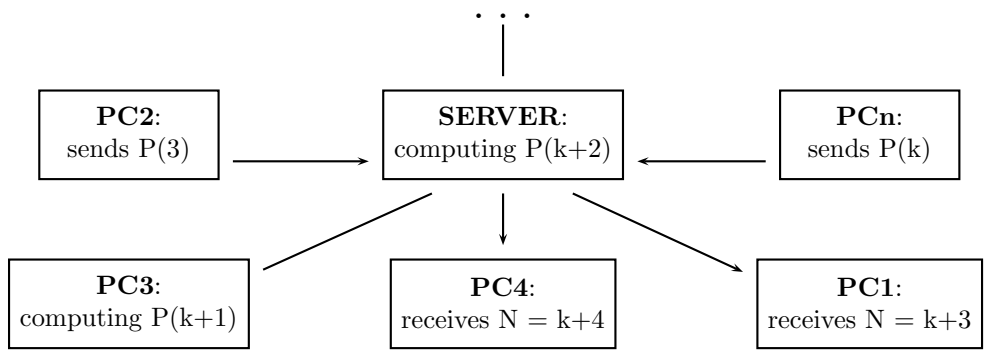


Fig. 6: the server sends new dilatation factors to the previous computers.

4. Some experimental results

As a first example, for $\lambda = (4, 3, 3, 2, 1)$, $\mu = (4, 3, 2, 2, 1)$ and $\nu = (7, 4, 4, 4, 3, 2, 1)$, an extract of the log file produced by the server application is following:

```
...
Tue Jun  7 17:15:47 2005  Received from "localhost": READY
Tue Jun  7 17:15:47 2005  Send "localhost": "compute f(1)"
Tue Jun  7 17:15:47 2005  Received from "PC_1": READY
Tue Jun  7 17:15:47 2005  Received from "localhost": LR-Coef. RESULT
Dilatation factor: N = 1  LR-Coef = 13 ; Computation time (s) = 0
Tue Jun  7 17:15:47 2005  Send "PC_1": "compute f(2)"
Tue Jun  7 17:15:47 2005  Received from "PC_2": READY
Tue Jun  7 17:15:47 2005  Received from "PC_1": LR-Coef. RESULT
Dilatation factor: N = 2  LR-Coef = 93 ; Computation time (s) = 0
...
Tue Jun  7 17:19:45 2005  Received from "localhost": LR-Coef. RESULT
Dilatation factor: N = 9  LR-Coef = 162019 ; Computation time (s) = 173
Tue Jun  7 17:31:01 2005  Received from "PC_3": LR-Coef. RESULT
Dilatation factor: N = 10 LR-Coef = 314743 ; Computation time (s) = 676
...
```

And here is the result of the interpolations:

```
Max. Deg :  8
Dilat.(N):  0    1     2     3      4      5        6        7        8
Coef.    :  1   13    93   456   1722   5382    14586    35376    78507
P1(N): 12N + 1
P2(N): 34N^2 - 22N + 1
P3(N): 1/6 (215N^3 - 441N^2 + 298N + 6)
P4(N): 1/24 (405N^4 - 1570N^3 + 2691N^2 - 1238N + 24)
P5(N): 1/120 (466N^5 - 2635N^4 + 8460N^3 - 9845N^2 + 4994N + 120)
P6(N): 1/360 (161N^6-1017N^5 + 5780N^4 - 10845N^3 + 14579N^2 - 4338N + 360)
P7(N): 1/5040 (123N^7-329N^6+7287N^5-9485N^4+47922N^3-12866N^2+27828N+5040)
P8(N) = 1/10080 (N^2 + 2N + 4)(5N + 21)(N + 1)(N + 2)(N + 3)(N + 4)(N + 5)
Total number of successive identical polynomials: 0
Real. Deg:  8
```

For the previous example, all the polynomials obtained by interpolation are different, which is experimentally not common. Here is another example which clearly points out the fact that the set of polynomials can be stationary:

```
Lambda: (7,6,5,4) ; Mu: (7,7,7,4) ; Nu: (12,8,8,7,6,4,2)
Max. Deg:   8
Dilat.(N):  0    1    2     3     4      5      6      7      8
Coef.    :  1   12   62   212   567   1288   2604   4824   8349
P1(N): 11N + 1
P2(N): 39/2 N^2 - 17/2 N + 1
P3(N): 61/6 N^3 - 11 N^2 + 71/6 N + 1
P4(N): 11/6 N^4 - 5/6 N^3 + 55/6 N^2 + 5/6 N + 1
P5(N): 1/30 (N+3)(N+2)(N+1)(3N^2+7N+5) = P6(N) = P7(N) = P8(N)
Total number of successive identical polynomials: 4
Real. Deg:  5
```

And below is a table showing some computations we've carried out. Those examples were arbitrarily selected.

	λ	μ	ν	max degree	real degree
1	9,7,3	9,9,3,2	10,9,9,8,6	4	1
2	2,2,1,1	2,2,1,1	3,3,2,2,1,1	4	2
3	11,10,8,5	20,17,3	26,25,8,8,7	4	3
4	6,5,2,2	5,5,3,2,2,2,1	8,8,7,7,2,2,1	5	0
5	9,8,3,3	10,7,5,3	10,10,8,8,7,5	5	2
6	11,10,8,5	20,17,3	26,25,8,8,5,2	5	4
7	9,5,3,3,3	7,6,5,4,3	10,10,8,8,7,5	5	5
8	11,8,8,5,2	20,17,3	26,25,8,8,5,2	5	5
9	5,4,3,3,2,1	9,5,3,3,2,1	9,8,8,6,5,5	6	6
10	18,11,9,4,2	20,17,9,4	26,25,19,16,8	6	6
11	64,30,27,17,9	55,48,32,12,4	84,75,66,49,24	6	6
12	5,3,2,2,1	4,3,2,2,1	7,5,4,4,3,2	7	7
13	7,6,5,4	7,7,7,4	12,8,8,7,6,4,2	8	5
14	4,3,3,2,1	4,3,3,2,1	7,4,4,4,3,2,1	8	8
15	5,5,3,2,2	6,6,4,2,1	10,6,6,5,5,2,2	9	4
16	11,10,8,4,2	8,7,6,5,2	18,17,15,7,4,2	10	2
17	5,4,4,3,3,2,1	9,6,4,3,2,2,1	9,9,8,8,6,5,4	10	10
18	5,4,4,3,3,2,1	9,7,3,3,2,2,1	10,9,8,7,6,5,4	11	0
19	4,4,2,2,1,1	4,4,2,2,1,1	8,8,3,3,2,2,1,1	12	2
20	5,5,3,2,1,1	6,6,4,2,1	6,6,6,5,5,3,3,2	12	3
21	5,5,3,2,1,1	6,6,4,2,1	6,6,6,5,5,3,2,2,1	14	5

Remark: examples 1, 10 and 11 in the table above are partitions taken from [2] (Fig. 3). With the exception of the first one, all the examples listed in that figure have a real degree equal to 6, for a predicted maximal degree also equal to 6.

5. Concluding remarks and perspectives

The examples listed show that the interpolation process very often produces some consecutive identical polynomials. In fact, all the computations made until now tend to show that the finite set of polynomials $(P_i{}^{\nu}_{\lambda\,\mu})$ is often stationary. If this result was proven, then it will be a fundamental supplementary information in the distribution of computations. As soon as the set $(P_i{}^{\nu}_{\lambda\,\mu})$ becomes stationary, the server will send an order to all computers, asking them to stop all computations still in progress. The sought-after polynomial being the last obtained by interpolation.

Translating the generated code from `Maple` into C increases the speed of computation of individual coefficients, as does the distribution of computations for the determination of $P^{\nu}_{\lambda\,\mu}$. The same techniques can be applied in the computation of many other combinatorial results, such as the generation of ribbon tableaux, spin and cospin polynomials. Such computations are being carried out at the LIPN[1], in collaboration with F. Descouens[2] who proposed [5] an algorithm for the generation of those combinatorial objects. Our application is still under development and is not yet stable enough. It will be available as part of `MuPAD-Combinat` [6] which is free and available online at `http://mupad-combinat.sourceforge.net`, after we have interfaced it with some other software like `LattE` [3] and `lrcalc` [1]. Finally, the hive model has been implemented using `Maple`, as well as the computation of $maxDeg$. We plan to convert those programs and include them in `MuPAD-Combinat`.

Acknowledgement: we would like to thank the JemSTIC[3] program for financing the participation of Frédéric Toumazet and Franck Butelle in this meeting. We would also like to thank the referees for their helpful remarks and suggestions.

Bibliography

1. *A.S Buch*, `http://home.imf.au.dk/abuch/lrcalc`
2. *C. Cochet*, *Vector partition and representation theory*, arXivmath.RT/0506159 V1, 9 Jun 2005.
3. *De Loera*, *J.A.*, *Haws*, *D.*, *Hemmecke*, *R.*, *Huggins*, *P.*, *Tauzer*, *J.*, *Yoshida*, *R. A User's Guide for LattE v1.1, 2003*, software package LattE is available at `http://www.math.ucdavis.edu/~latte/`
4. *E. Rassart*, *A polynomiality property for Littlewood-Richardson coefficients*. J Combinatorial Theory, Ser A 107 (2004) 161-179.
5. *F. Descouens*, *Un algorithme de génération des tableaux de rubans et de calcul de polynôme de spin*, Journées Montoises, Proceedings, Liège, 2004.
6. *F.Hivert* and *N.Thiery*, *MuPAD-Combinat, an Open Source Package for Research in Algebraic Combinatorics*, Séminaire Lotharingien de Combinatoire 51 (2004).
7. *H. Derksen* and *J. Weyman*, *On the Littlewood-Richardson polynomials*, J. Algebra, 255(2002) 247-257.
8. *I. G. MacDonald*, *Symmetric functions and Hall Polynomials*, 2nd ed., Clarendon Press, Oxford Science Publications, 1995.
9. *R. C. King*, *C. Tollu*, and *F. Toumazet*, *Stretched Littlewood-Richardson and Kostka coefficients*, CRM Proceedings and Lecture Notes, Vol. 34, Amer. Math. Soc, 2004, pp99-112.
10. *R. C. King*, *C. Tollu*, and *F. Toumazet*, *The hive model and the polynomial nature of stretched Littlewood-Richardson coefficients*. Presented at FPSAC'05.

[1]Laboratoire d'Informatique Paris Nord, Avenue J.B. Clément, 93430 Villetaneuse, FRANCE
[2]Ph.D. student, Université de Marne-la-Vallée, *francois.descouens@univ-mlv.fr*
[3]"Action Jeune Equipe et Mobilité" - CNRS