Computer algebra algorithms of simplification of tensor expressions

A.Kryukov[1], G.Shpiz

SINP MSU, Msocow

[1]E-mail: kryukov@theory.sinp.msu.ru

Outlook

- **Introduction**
- **Problem**
- **Permutation group and tensor symmetry**
- **Dummy (summation) indices**
- **Algorithms**
  - Double coset
  - Group algebra
  - Graph isomorphism
- **Conclusions**

Introduction

- **Tensor calculation is widely use in**
  - Theoretical physics
  - Solid state physics
  - Mechanics
- **We will discuss different algorithms for simplification of tensor expressions**
- **We also presented a sketch of algorithm based on an isomorphism of graphs corresponding to tensor monomials**
- **Types of tensor calculations**
  - Components calculations
  - Tensor with abstract index calculations
  - Abstract tensor calculations

Introduction: Component calculations

- **Choose a coordinate basis**
- **Calculation of components as scalar objects**
- **Advantages:**
  - Often we need the value of tensor components in fixed coordinate bases
- **Disadvantages:**
  - Does not utilize specific properties of tensors like symmetries
- **We do not discuss this approach**

Introduction: Tensor with abstract indices

- **T(i,j,k,l) – as an abstract indexed object**
- **We do not use a fixed coordinate basis**
  - May use knowledge about the dimension of the corresponding space
- **Advantage:**
  - Use symmetry properties with respect to permutation of indices and summation indices too
- **Disadvantage:**
  - Good for calculation of invariants
  - We often need the value of components at the end of calculation too

Introduction: Abstract tensor expressions

- **Just for illustration**
  - Exterior algebra
- **Not discuss here at all**

Problem: Simplification of tensor expressions

- **For simplicity we will not distinguish the upper and lower indices**
- **We also do not interested in the transformation properties of tensors under coordinate transformations**
- **We will consider tensor as an indexed formal term which may have defined symmetry properties.**
- **Let us consider the tensor expression:**

  T(i,j,k,l)*T(k,l,m,n)+…
- **There are two main problems:**
  - Are the two monomials equal?
  - Is a monomial equal to zero?

Permutation group and tensor symmetry

- **Let us consider the Riemann tensor:**

  R(i,j,k,l)=-R(j,i,k,l)

  R(i,j,k,l)=R(k,l,i,j)

- **Let π be an element of the permutation group S_4**

  ∃ π∈S(4): π(1,2,3,4)=(2,1,3,4)

  R(i,j,k,l)=(-1)*R(π(i,j,k,l))

- **Thus the symmetries of the (Riemann) tensor form a subgroup of the permutation group**

Dummy (summation) indices

- **Ricci tensor:**

  $R(j,l)=R(i,j,i,l)$

- **Scalar curvature:**

  $R=R(m,m)=R(i,m,i,m)$

- **Renaming of dummies:**

  $R(i,m,i,m)=R(m,i,m,i)$

- **Permutation dummies:**

  $R(i_1,m,i_2,m)=R(i_2,m,i_1,m)$

Algorithm: Double coset

- **Symmetry group (S)**
  - Subgroup of permutation group acting from the right
- **Dummy indices (L)**
  - subgroup of permutation group acting from the left
- **Simplification problem is equivalent to finding double coset**

  L\T/S
- **See details**

  A.Rodionov, A.Taranov, EUROCAL’87, LNCS, vol. 378 (1989) p. 192.

  G.Butler, LNCS, vol.559 (1991)

  L.R.U.Manssur, R.Portugal, CPC, vol.157(2004), p.173
- **Problem: multiterm identity (like Bianci identity)**

Algorithm: Group algebra

- **Let us consider the group algebra**

  $t_1 * e^1 + t_2 * e^2 + \ldots$ (n! Terms)

  where $e_i = T(\pi_i(1,2,3,4))$, $i=1..n!$

- **All available relations define a k-dimensional hyperplane K in $R^{n!}$ Space:**

  $R^{n!} = K + Q$

- **All tensor relations can be treat in a unified manner**
- **The conjugate space Q is the space of canonical elements**
- **See details**

  V.Ilyin, A.Kryukov, CPC, vol. 96, No 1, pp 36-52

- **Problem: Dimension of space is n!**

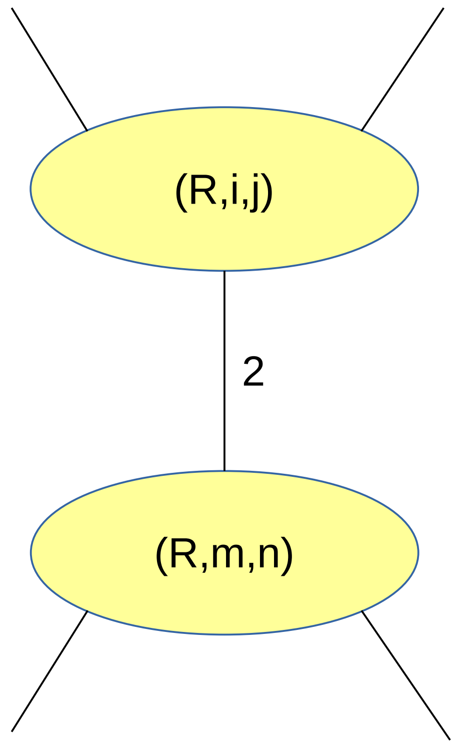
Algorithm: Mean over the orbit of monomial

- **In practical cases we have:**
  - the monomials have not so many tensor terms (~10)
  - Each term has rather reach symmetries
    - R(i,j,k,l): 4!=24 but the only 2 independent components
- **If the number of terms in a monomial is about 5 the simplest way to find the stabilizer of the monomial is calculation of the average over the orbit of the monomial**

Algorithm: Graph isomorphism

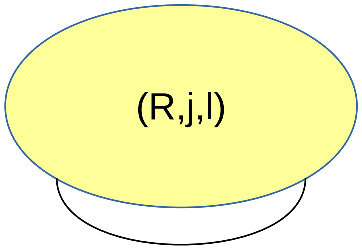
- **Tensor type and the set of indices without summation will be called tensor signature**
- **Each monomial maps to a colored graph**
- **A vertex is a tensor in the monomial. The color of the vertex is defined by its signature.**
- **The internal edges correspond to the summation of indices with a weight**
- **Example:**

  R(i,j,k,l)*R(k,l,m,n) →
  R(i,j,k,l)*R(m,n,k,l)



Algorithm: Graph isomorthism

- **Lexicographical ordering:**
  - T_1(i,j,..) < T_2(n,m,..) iff T_1 and T_2 have the same type and (i,j,..) ≤ (m,n,..)
  - Dummies > any free indices
- **Transform each tensor to the canonical form with respect to its symmetry properties.**
  - There is a special simplification procedure for each tensor type
  - No internal contractions of indices. Otherwise this is another type of tensor (Riemann → Ricci)



Algorithm: Graph isomorphism

- **Every step should be finished by the canonization of tensor itself and ordering the tensor in monomials (so called “pre-canonical form”)**
- **Apply the allowed transformation of dummy indices to the monomial.**
  - For example exchange two pairs of dummies
- **Calculate the average over all the monomials which have isomorphic graphs with the initial one**
- **The obtained polynomial is invariant with respect to the transformations from the equivalence class of the initial monomial**

Algorithm: Graph isomorphism

- **Using the obtained invariants we can get the canonical form of the monomials**
  - Two monomials is equal iff their canonical forms is the same lexicographically
- **The algorithm significantly reduce the amount of computations in the case of large groups.**
  - Every time we work among orbits which not so many.
- **Similar approaches:**
  - Zhendong Li, Sihong Shao, Wenjian Liu, arXiv:1604.06156v1
  - S.Poslavsky, D.Bolotin, Journal of Physics: Conference Series, Vol.608

Conclusions

- **Now we realize a prototype of the program on Python language**
- **The program is working rather good for tensor expressions in practical cases where monomials contain 10-20 terms.**
- **Each term contains up to 10 indices**
- **We have a plan to optimize the program and to compare it with other similar programs**

Thank you very much

Questions?