Formal proofs of hypergeometric sums
Dedicated to the memory of Andrzej Trybulec

John Harrison

Intel Corporation

1st July 2014 (11:10 – 12:00)

Overview

Overview

- Some memories of Andrzej, Bialystok and Cambridge

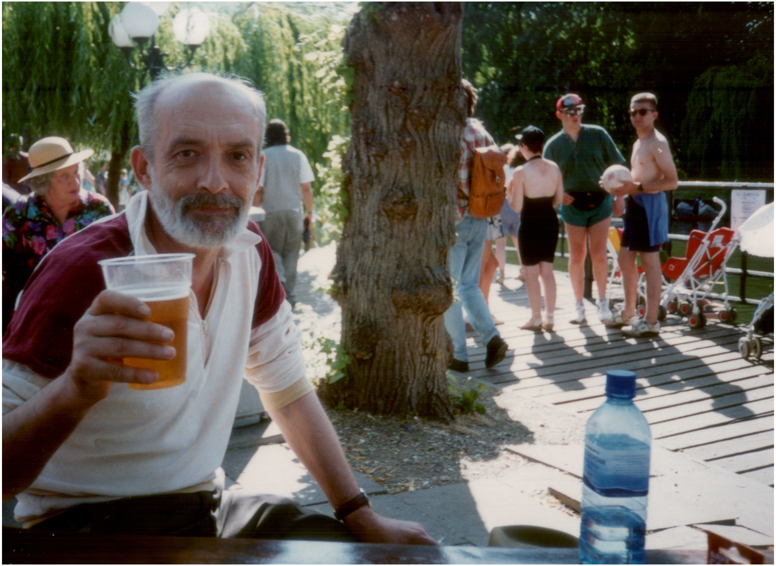
Overview

- Some memories of Andrzej, Bialystok and Cambridge
- ☺ Trouble-free formalization: some topological theorems due to Borsuk.

Overview

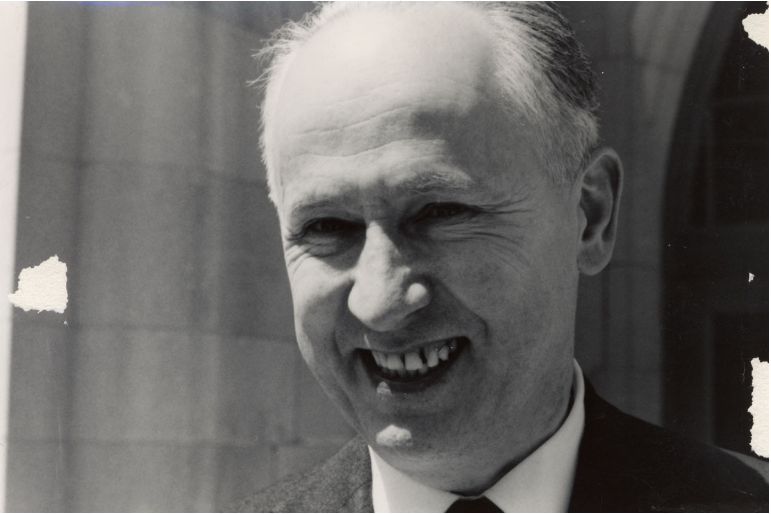
- Some memories of Andrzej, Bialystok and Cambridge
- ☺ Trouble-free formalization: some topological theorems due to Borsuk.
- ☹ Problematic formalization: Wilf-Zeilberger method for hypergeometric summation.
    - Hypergeometric sequences
    - Gosper's algorithm and the WZ method
    - WZ examples, and their difficulties
    - Generic proof of Sylvester's identity, limit formulation of WZ
    - Formalizing the gamma function
    - Avoiding a countable family of algebraic varieties
    - The method at work
    - Automation and conclusions

Memories of Andrzej



Memories of Andrzej

Back to Borsuk . . .



The Borsuk homotopy extension theorem

Fundamental in relating homotopy to extension properties:

```
BORSUK_HOMOTOPY_EXTENSION_HOMOTOPIC =
 |- !f:real^M->real^N g s t u.
       closed_in (subtopology euclidean t) s /\
       (ANR s /\ ANR t \/ ANR u) /\
       f continuous_on t /\
       IMAGE f t SUBSET u /\
       homotopic_with (\x. T) (s,u) f g
       ==> ?g'. homotopic_with (\x. T) (t,u) f g' /\
                g' continuous_on t /\
                IMAGE g' t SUBSET u /\
                !x. x IN s ==> g'(x) = g(x)
```

Bosuk's separation theorem

Characterize separation properties in purely homotopic terms

```
BORSUK_SEPARATION_THEOREM_GEN =
 |- !s:real^N->bool.
     compact s
     ==> ((!c. c IN components((:real^N) DIFF s)
               ==> ~bounded c) <=>
          (!f. f continuous_on s /\
               IMAGE f s SUBSET sphere(vec 0,&1)
             ==> ?c. homotopic_with (\x. T)
                     (s,sphere(vec 0,&1)) f (\x. c)))
```

Note that the $N = 1$ case is a bit different, but this statement works uniformly there too.

Separating space is a homotopy invariant

For compact sets, whether they separate space or not respects homotopy equivalence

```
HOMOTOPY_EQUIVALENT_SEPARATION =
 |- !s t. compact s /\ compact t /\
          s homotopy_equivalent t
          ==> (connected((:real^N) DIFF s) <=>
               connected((:real^N) DIFF t))
```

Separating space is a homotopy invariant

For compact sets, whether they separate space or not respects homotopy equivalence

```
HOMOTOPY_EQUIVALENT_SEPARATION =
 |- !s t. compact s /\ compact t /\
          s homotopy_equivalent t
          ==> (connected((:real^N) DIFF s) <=>
               connected((:real^N) DIFF t))
```

This yields in particular a major part of the Jordan Curve Theorem in a more general context

```
JORDAN_BROUWER_SEPARATION =
 |- !s a:real^N r.
        &0 < r /\
        s homeomorphic sphere(a,r)
        ==> ~connected((:real^N) DIFF s)
```

$A = B$

There are algorithmic symbolic methods that can often do a remarkably good job of automating the proof (or discovery) of quite complicated sums.

$A = B$

There are algorithmic symbolic methods that can often do a remarkably good job of automating the proof (or discovery) of quite complicated sums.

- Gosper's algorithm for hypergeometric antidifferences

$A = B$

There are algorithmic symbolic methods that can often do a remarkably good job of automating the proof (or discovery) of quite complicated sums.

- Gosper's algorithm for hypergeometric antidifferences
- Zeilberger's general method using closure properties of holonomic sequences

$A = B$

There are algorithmic symbolic methods that can often do a remarkably good job of automating the proof (or discovery) of quite complicated sums.

- Gosper's algorithm for hypergeometric antidifferences
- Zeilberger's general method using closure properties of holonomic sequences
- Wilf-Zeilberger method

$A = B$

There are algorithmic symbolic methods that can often do a remarkably good job of automating the proof (or discovery) of quite complicated sums.

- Gosper's algorithm for hypergeometric antidifferences
- Zeilberger's general method using closure properties of holonomic sequences
- Wilf-Zeilberger method

We are mainly interested in formalizing WZ results, but we also discuss Gosper's algorithm since it's an essential component of WZ.

$A = B$

There are algorithmic symbolic methods that can often do a remarkably good job of automating the proof (or discovery) of quite complicated sums.

- Gosper's algorithm for hypergeometric antidifferences
- Zeilberger's general method using closure properties of holonomic sequences
- Wilf-Zeilberger method

We are mainly interested in formalizing WZ results, but we also discuss Gosper's algorithm since it's an essential component of WZ.

Reference: '$A = B$' by Marko Petkovšek, Herbert S. Wilf and Doron Zeilberger.

Hypergeometric sequences

A *hypergeometric sequence* (or *term* or *series*) is one where the ratio of successive terms is a rational function of n.

$$a_{n+1}/a_n = r(n) = p(n)/q(n)$$

For example, factorials where $(n+1)!/n! = n+1$, the 'power of 2' function with $2^{n+1}/2^n = 2$.

Hypergeometric sequences

A *hypergeometric sequence* (or *term* or *series*) is one where the ratio of successive terms is a rational function of n.

$$a_{n+1}/a_n = r(n) = p(n)/q(n)$$

For example, factorials where $(n+1)!/n! = n+1$, the 'power of 2' function with $2^{n+1}/2^n = 2$.
We call a function of several variables hypergeometric if it's hypergeometric in each argument separately, e.g. binomial coefficients

$$\binom{n+1}{k} = \frac{n+1}{n-k+1}\binom{n}{k}$$

$$\binom{n}{k+1} = \frac{n-k}{k+1}\binom{n}{k}$$

Gosper's algorithm

Given a hypergeometric term t_k, Gosper's algorithm will either

Gosper's algorithm

Given a hypergeometric term t_k, Gosper's algorithm will either

- Find a hypergeometric 'antidifference' or 'indefinite sum' s_k such that $s_{k+1} - s_k = t_k$

Gosper's algorithm

Given a hypergeometric term t_k, Gosper's algorithm will either

- Find a hypergeometric 'antidifference' or 'indefinite sum' s_k such that $s_{k+1} - s_k = t_k$
- Determine that *no such hypergeometric antidifference exists*

Gosper's algorithm

Given a hypergeometric term t_k, Gosper's algorithm will either

- Find a hypergeometric 'antidifference' or 'indefinite sum' s_k such that $s_{k+1} - s_k = t_k$
- Determine that *no such hypergeometric antidifference exists*

An antidifference also lets us solve *definite* summation problems:

$$\sum_{k=a}^{b} t_k = \sum_{k=a}^{b} (s_{k+1} - s_k) = s_{b+1} - s_a$$

Gosper's algorithm

Given a hypergeometric term t_k, Gosper's algorithm will either

- Find a hypergeometric 'antidifference' or 'indefinite sum' s_k such that $s_{k+1} - s_k = t_k$
- Determine that *no such hypergeometric antidifference exists*

An antidifference also lets us solve *definite* summation problems:

$$\sum_{k=a}^{b} t_k = \sum_{k=a}^{b} (s_{k+1} - s_k) = s_{b+1} - s_a$$

If $s_{k+1} - s_k = t_k$ and s_k is hypergeometric, s_k and t_k are rational-function multiples of each other, so t_k is hypergeometric too.

Gosper's algorithm

Given a hypergeometric term t_k, Gosper's algorithm will either

- Find a hypergeometric 'antidifference' or 'indefinite sum' s_k such that $s_{k+1} - s_k = t_k$
- Determine that *no such hypergeometric antidifference exists*

An antidifference also lets us solve *definite* summation problems:

$$\sum_{k=a}^{b} t_k = \sum_{k=a}^{b} (s_{k+1} - s_k) = s_{b+1} - s_a$$

If $s_{k+1} - s_k = t_k$ and s_k is hypergeometric, s_k and t_k are rational-function multiples of each other, so t_k is hypergeometric too.
However some hypergeometric terms have no hypergeometric antidifference.

Gosper example

Consider the term $t_k = \frac{k \cdot k!}{n^k}\binom{n}{k}$. We'll use the implementation of Gosper's algorithm in Maxima due to Fabrizio Caruso:

Gosper example

Consider the term $t_k = \frac{k \cdot k!}{n^k}\binom{n}{k}$. We'll use the implementation of Gosper's algorithm in Maxima due to Fabrizio Caruso:

```
(%i2) AntiDifference(k * k! * binomial(n,k) / n^k,k);
                                         1 - k
(%o2)                           - k! n        binomial(n, k)
```

Gosper example

Consider the term $t_k = \frac{k \cdot k!}{n^k}\binom{n}{k}$. We'll use the implementation of Gosper's algorithm in Maxima due to Fabrizio Caruso:

```
(%i2) AntiDifference(k * k! * binomial(n,k) / n^k,k);
                                          1 - k
(%o2)                          - k! n          binomial(n, k)
```

That is $s_k = -k!n^{1-k}\binom{n}{k}$. This lets us easily verify the following *definite* sum, which was problem E 3088 in the "American Mathematical Monthly".

$$\sum_{k=1}^{n} \frac{k \cdot k!}{n^k}\binom{n}{k} = s_{n+1} - s_1 = n$$

From Gosper to WZ

We'll explicitly consider terms parametrized by n, say $F(n, k)$ where summation is over k, with finite support w.r.t. k for each n.

From Gosper to WZ

We'll explicitly consider terms parametrized by n, say $F(n, k)$ where summation is over k, with finite support w.r.t. k for each n. Even when a hypergeometric term has a hypergeometric *definite* sum, it might not have a hypergeometric *antidifference*, so Gosper's algorithm doesn't help, e.g.

From Gosper to WZ

We'll explicitly consider terms parametrized by n, say $F(n, k)$ where summation is over k, with finite support w.r.t. k for each n. Even when a hypergeometric term has a hypergeometric *definite* sum, it might not have a hypergeometric *antidifference*, so Gosper's algorithm doesn't help, e.g.

$$\sum_{k=0}^{n} \binom{n}{k} = \sum_{k=0}^{n} \binom{n}{k} 1^k 1^{n-k} = (1+1)^n = 2^n$$

but it turns out $\binom{n}{k}$ has no hypergeometric *antidifference*.

From Gosper to WZ

We'll explicitly consider terms parametrized by n, say $F(n,k)$ where summation is over k, with finite support w.r.t. k for each n. Even when a hypergeometric term has a hypergeometric *definite* sum, it might not have a hypergeometric *antidifference*, so Gosper's algorithm doesn't help, e.g.

$$\sum_{k=0}^{n} \binom{n}{k} = \sum_{k=0}^{n} \binom{n}{k} 1^k 1^{n-k} = (1+1)^n = 2^n$$

but it turns out $\binom{n}{k}$ has no hypergeometric *antidifference*. The idea of the WZ algorithm is to apply Gosper not to $F(n,k)$ itself, but rather to $F(n+1,k) - F(n,k)$ (or in general a more complicated combination, but we'll ignore that here).

The basic WZ idea

We say that $G(n,k)$ is the WZ-mate of $F(n,k)$, and that F and G form a 'WZ-pair', when

$$F(n+1,k) - F(n,k) = G(n,k+1) - G(n,k)$$

The basic WZ idea

We say that $G(n,k)$ is the WZ-mate of $F(n,k)$, and that F and G form a 'WZ-pair', when

$$F(n+1,k) - F(n,k) = G(n,k+1) - G(n,k)$$

We get a similar telescoping phenomenon summing over k

$$\sum_{k=a}^{b} F(n+1,k) - \sum_{k=a}^{b} F(n,k) = G(n,b+1) - G(n,a)$$

The basic WZ idea

We say that $G(n,k)$ is the WZ-mate of $F(n,k)$, and that F and G form a 'WZ-pair', when

$$F(n+1,k) - F(n,k) = G(n,k+1) - G(n,k)$$

We get a similar telescoping phenomenon summing over k

$$\sum_{k=a}^{b} F(n+1,k) - \sum_{k=a}^{b} F(n,k) = G(n,b+1) - G(n,a)$$

If $G(n,k)$ has finite support, summing over all (or enough) integers shows $\sum_k F(n+1,k) - \sum_k F(n,k) = 0$, i.e. the sum is *independent of* n.

The WZ method

If we want to verify a summation of the form $\sum_k F(n,k) = S(n)$

The WZ method

If we want to verify a summation of the form $\sum_k F(n,k) = S(n)$

1. Divide through by $S(n)$ so we just need the special case $\sum_k F(n,k) = 1$

The WZ method

If we want to verify a summation of the form $\sum_k F(n,k) = S(n)$

1. Divide through by $S(n)$ so we just need the special case $\sum_k F(n,k) = 1$
2. Apply Gosper's algorithm to find a WZ-mate $G(n,k)$ with $F(n+1,k) - F(n,k) = G(n,k+1) - G(n,k)$

The WZ method

If we want to verify a summation of the form $\sum_k F(n,k) = S(n)$

1. Divide through by $S(n)$ so we just need the special case $\sum_k F(n,k) = 1$
2. Apply Gosper's algorithm to find a WZ-mate $G(n,k)$ with $F(n+1,k) - F(n,k) = G(n,k+1) - G(n,k)$
3. Conclude that $\sum_k F(n,k)$ is independent of n and so we just need to check the following, which we expect to be easy $\sum_k F(0,k) = 1$

WZ example

Closely following '$A = B$', we prove $\sum_k \binom{n}{k}^2 = \binom{2n}{n}$

WZ example

Closely following '$A = B$', we prove $\sum_k \binom{n}{k}^2 = \binom{2n}{n}$

1. We divide through by the right-hand side so we need to verify $\sum_k F(n,k) = 1$ where

$$F(n,k) = \frac{\binom{n}{k}^2}{\binom{2n}{n}} = \frac{n!^4}{k!^2(n-k)!^2(2n)!}$$

WZ example

Closely following '$A = B$', we prove $\sum_k \binom{n}{k}^2 = \binom{2n}{n}$

1. We divide through by the right-hand side so we need to verify $\sum_k F(n,k) = 1$ where

$$F(n,k) = \frac{\binom{n}{k}^2}{\binom{2n}{n}} = \frac{n!^4}{k!^2(n-k)!^2(2n)!}$$

2. We apply Gosper's algorithm to obtain the magic rational function

$$R(n,k) = \frac{-k^2(3n-2k+3)}{2(n-k+1)^2(2n+1)}$$

such that $G(n,k) = R(n,k)F(n,k)$ satisfies the key property $F(n+1,k) - F(n,k) = G(n,k+1) - G(n,k)$

WZ example

Closely following '$A = B$', we prove $\sum_k \binom{n}{k}^2 = \binom{2n}{n}$

1. We divide through by the right-hand side so we need to verify $\sum_k F(n,k) = 1$ where

$$F(n,k) = \frac{\binom{n}{k}^2}{\binom{2n}{n}} = \frac{n!^4}{k!^2(n-k)!^2(2n)!}$$

2. We apply Gosper's algorithm to obtain the magic rational function

$$R(n,k) = \frac{-k^2(3n-2k+3)}{2(n-k+1)^2(2n+1)}$$

such that $G(n,k) = R(n,k)F(n,k)$ satisfies the key property $F(n+1,k) - F(n,k) = G(n,k+1) - G(n,k)$

3. So $\sum_k F(n,k)$ is independent of n, so we can evaluate the case $n = 0$, which is easy to simplify to 1

A routine formalization?

It now seems very natural to formally certify the basic manipulations, using a standard WZ implementation to provide the key rational function. All we need is to formalize this:

A routine formalization?

It now seems very natural to formally certify the basic manipulations, using a standard WZ implementation to provide the key rational function. All we need is to formalize this:

> *"Well, at this point we have arrived at a situation that will be referred to throughout this book as a "routinely verifiable" identity. That phrase means roughly that your pet chimpanzee could check out the equation. More precisely it means this. First cancel out all factors that look like c^n or c^k [...] that can be cancelled. Then replace every binomial coefficient in sight by the quotient of factorials that it represents. Finally, cancel out all of the factorials by suitable divisions, leaving only a polynomial identity that involves n and k." (from '$A = B$')*

Problems

Unfortunately when you examine it closely (as formalizing makes you do) this looks much too glib:

Problems

Unfortunately when you examine it closely (as formalizing makes you do) this looks much too glib:

- We are supposed to sum over *all integers*, but how are factorials and binomial coefficients defined for negative numbers?

Problems

Unfortunately when you examine it closely (as formalizing makes you do) this looks much too glib:

- We are supposed to sum over *all integers*, but how are factorials and binomial coefficients defined for negative numbers?
- It's not at all clear why it's valid to replace $\binom{n}{k} = n!/k!(n-k)!$ since that is in general only valid/meaningful for $0 \leq k \leq n$

Problems

Unfortunately when you examine it closely (as formalizing makes you do) this looks much too glib:

- We are supposed to sum over *all integers*, but how are factorials and binomial coefficients defined for negative numbers?
- It's not at all clear why it's valid to replace $\binom{n}{k} = n!/k!(n-k)!$ since that is in general only valid/meaningful for $0 \leq k \leq n$
- The rational function certificates often have poles, so on the face of it we seem to be simplifying terms of the form $0/0$.

Problems

Unfortunately when you examine it closely (as formalizing makes you do) this looks much too glib:

- We are supposed to sum over *all integers*, but how are factorials and binomial coefficients defined for negative numbers?
- It's not at all clear why it's valid to replace $\binom{n}{k} = n!/k!(n-k)!$ since that is in general only valid/meaningful for $0 \leq k \leq n$
- The rational function certificates often have poles, so on the face of it we seem to be simplifying terms of the form $0/0$.

It seems very hard to avoid these issues in a nice and automatable way if we use a straightforward interpretation.

Generic proof of Sylvester's identity

For inspiration we look at the proof in HOL Light of Sylvester's determinant identity $\det(I + AB) = \det(I + BA)$.

Generic proof of Sylvester's identity

For inspiration we look at the proof in HOL Light of Sylvester's determinant identity $\det(I + AB) = \det(I + BA)$.
It's fairly easy by padding out the matrices to assume they are square. Then we have

$$\det(I + AB)\det(A) = \det(A + ABA) = \det(A)\det(I + BA)$$

and the result follows by cancelling $\det(A)$

Generic proof of Sylvester's identity

For inspiration we look at the proof in HOL Light of Sylvester's determinant identity $\det(I + AB) = \det(I + BA)$.
It's fairly easy by padding out the matrices to assume they are square. Then we have

$$\det(I + AB)\det(A) = \det(A + ABA) = \det(A)\det(I + BA)$$

and the result follows by cancelling $\det(A)$ *provided that is not zero*.

Generic proof of Sylvester's identity

For inspiration we look at the proof in HOL Light of Sylvester's determinant identity $\det(I + AB) = \det(I + BA)$.

It's fairly easy by padding out the matrices to assume they are square. Then we have

$$\det(I + AB)\det(A) = \det(A + ABA) = \det(A)\det(I + BA)$$

and the result follows by cancelling $\det(A)$ *provided that is not zero*.

To handle the general case we use a *limit* argument, that every matrix can be approached arbitrarily closely by an invertible one.

This effectively lets us choose a 'generic' matrix in the main argument.

We want to do the same sort of thing with WZ.

The gamma function

In order to use limits, we need to generalize things from the integers to the reals, defining $\Gamma(z)$ such that $\Gamma(n+1) = n!$

Formalizing the gamma function

We define complex gamma functions via the following limit, though we derive other equivalent forms of the definition

$$\Gamma(z) = \lim_{n\to\infty} \frac{n^z n!}{\prod_{m=0}^{n}(z+m)}$$

Formalizing the gamma function

We define complex gamma functions via the following limit, though we derive other equivalent forms of the definition

$$\Gamma(z) = \lim_{n\to\infty} \frac{n^z\, n!}{\prod_{m=0}^{n}(z+m)}$$

In HOL Light:

```
|- cgamma(z) =
   lim sequentially
    (\n. (Cx(&n) cpow z * Cx(&(FACT n))) /
         cproduct(0..n) (\m. z + Cx(&m)))
```

We derive many useful properties and specialize to the *real* gamma function `gamma`, which is what we use here.

Generalizing to the reals

We establish some definitions to generalize factorials and binomial coefficients to the reals:

```
|- rfact x = gamma(x + &1)

|- rbinom(n,k) = rfact n / (rfact k * rfact (n - k))
```

Generalizing to the reals

We establish some definitions to generalize factorials and binomial coefficients to the reals:

```
|- rfact x = gamma(x + &1)

|- rbinom(n,k) = rfact n / (rfact k * rfact (n - k))
```

In general, factorials are still not well-defined at negative integers, and similarly not all binomial coefficients make sense.

Generalizing to the reals

We establish some definitions to generalize factorials and binomial coefficients to the reals:

```
|- rfact x = gamma(x + &1)

|- rbinom(n,k) = rfact n / (rfact k * rfact (n - k))
```

In general, factorials are still not well-defined at negative integers, and similarly not all binomial coefficients make sense.
But they behave very well as limits

```
|- !net nn kk n k.
        (nn ---> &n) net /\ (kk ---> &k) net
        ==> ((\a. rbinom(nn a,kk a))
                ---> &(binom(n,k))) net
```

This lets us justify all the 'naive' manipulations in this context without any case analysis.

Making limits work

However, to make the limit argument work, we need to show we can approach a pair (n, k) arbitrarily closely while avoiding various special values:

Making limits work

However, to make the limit argument work, we need to show we can approach a pair (n, k) arbitrarily closely while avoiding various special values:

- Those where Γ is applied to negative integers where it is undefined, or where recurrence formulas fail.

Making limits work

However, to make the limit argument work, we need to show we can approach a pair (n, k) arbitrarily closely while avoiding various special values:

- Those where Γ is applied to negative integers where it is undefined, or where recurrence formulas fail.
- Those where the denominator of the rational function in the certificate becomes zero.

Making limits work

However, to make the limit argument work, we need to show we can approach a pair (n, k) arbitrarily closely while avoiding various special values:

- Those where Γ is applied to negative integers where it is undefined, or where recurrence formulas fail.
- Those where the denominator of the rational function in the certificate becomes zero.

All problem cases (x, y) are defined by a bivariate polynomial with rational coefficients.

Making limits work

However, to make the limit argument work, we need to show we can approach a pair (n, k) arbitrarily closely while avoiding various special values:

- Those where Γ is applied to negative integers where it is undefined, or where recurrence formulas fail.
- Those where the denominator of the rational function in the certificate becomes zero.

All problem cases (x, y) are defined by a bivariate polynomial with rational coefficients.

```
|- ratpolyfun p <=>
        ?s. FINITE s /\
            s SUBSET (:num#num) CROSS rational /\
            p = \(x,y). sum s
                  (\((i,j),c). c * x pow i * y pow j)

|- ratty t <=>
     ?p. ratpolyfun p /\ p t = &0 /\ ~(!w. p w = &0)
```

Avoiding a countable family of algebraic varieties

We want to show that any integer point (n, k) can be approached arbitrarily closely by a pair of reals (x, y) that is not 'ratty'. This follows from:

Avoiding a countable family of algebraic varieties

We want to show that any integer point (n, k) can be approached arbitrarily closely by a pair of reals (x, y) that is not 'ratty'. This follows from:

A non-trivial algebraic variety has empty interior.

```
|- !f c. real_polynomial_function f /\
         ~(!x. f x = c)
         ==> interior {x | f(x) = c} = {}
```

Avoiding a countable family of algebraic varieties

We want to show that any integer point (n, k) can be approached arbitrarily closely by a pair of reals (x, y) that is not 'ratty'. This follows from:

A non-trivial algebraic variety has empty interior.

```
|- !f c. real_polynomial_function f /\
         ~(!x. f x = c)
         ==> interior {x | f(x) = c} = {}
```

A countable union of nowhere dense sets has empty interior (this is a Baire-type result):

```
|- !g:(real^N->bool)->bool.
       COUNTABLE g /\
       (!s. s IN g ==> closed s /\ interior s = {})
       ==> interior(UNIONS g) = {}
```

The WZ limit theorem

Hence we can obtain a WZ-type theorem that allows one free rein to manipulate terms ‘naively’:

```
|- (!n. FINITE {k | ~(f n k = &0)}) /\
   (!n k. (FF ---> f n k) (at(complex(&n,&k)))) /\
   (!n k. integer k /\ k < &0
          ==> (FF ---> &0) (at(complex(&n,k)))) /\
   (!n B. ?i j. integer i /\ integer j /\
                i < &0 /\ &B <= j /\
                (GG ---> &0) (at(complex(&n,i))) /\
                (GG ---> &0) (at(complex(&n,j)))) /\
   (!n k. ~ratty(n,k)
          ==> FF(complex(n + &1,k)) -
              FF(complex(n,k)) =
              GG(complex(n,k + &1)) -
              GG(complex(n,k))) /\
   sum (:num) (f 0) = 1
   ==> !n. sum (:num) (f n) = 1
```

An example

We define appropriate $F(n, k)$ and $G(n, k)$ for the example $\sum_{k=0}^{n} \binom{n}{k} = 2^n$, as functions $\mathbb{C} \to \mathbb{R}$:

```
|- FF z = rbinom(z$1,z$2) / &2 rpow z$1

|- RR z = z$2 / (&2 * (z$2 - z$1 - &1))
```

An example

We define appropriate $F(n,k)$ and $G(n,k)$ for the example $\sum_{k=0}^{n} \binom{n}{k} = 2^n$, as functions $\mathbb{C} \to \mathbb{R}$:

```
|- FF z = rbinom(z$1,z$2) / &2 rpow z$1

|- RR z = z$2 / (&2 * (z$2 - z$1 - &1))
```

We can justify the WZ-pair property for *all reals* except for a few special cases, all 'ratty'

```
|- ~(n + &1 = &0) /\ ~(n + &1 = k) /\
   ~(k + &1 = &0) /\ ~(n = k)
   ==> FF(complex(n + &1,k)) - FF(complex(n,k)) =
       GG(complex(n,k + &1)) - GG(complex(n,k))
```

Conclusions and thoughts on automation

- We have tested examples by hand and they all work exactly following the style of the 'naive' presentation in '$A = B$', but now justified by a clear semantics.

Conclusions and thoughts on automation

- We have tested examples by hand and they all work exactly following the style of the 'naive' presentation in '$A = B$', but now justified by a clear semantics.
- Most of the algebraic manipulations are quite easy to automate, including the cancellation of factorials, since we avoid any possible trouble points or singularities.

Conclusions and thoughts on automation

- We have tested examples by hand and they all work exactly following the style of the 'naive' presentation in '$A = B$', but now justified by a clear semantics.
- Most of the algebraic manipulations are quite easy to automate, including the cancellation of factorials, since we avoid any possible trouble points or singularities.
- There is still currently a bit of manual work involved in showing that suitable limits for end values of G exist.

Conclusions and thoughts on automation

- We have tested examples by hand and they all work exactly following the style of the ‘naive’ presentation in ‘$A = B$’, but now justified by a clear semantics.
- Most of the algebraic manipulations are quite easy to automate, including the cancellation of factorials, since we avoid any possible trouble points or singularities.
- There is still currently a bit of manual work involved in showing that suitable limits for end values of G exist.
- However, this could be automated too by arguing that the denominator of the certificate, for fixed n, is a polynomial in k and hence is nonzero for large enough $|k|$.

Conclusions and thoughts on automation

- We have tested examples by hand and they all work exactly following the style of the 'naive' presentation in '$A = B$', but now justified by a clear semantics.
- Most of the algebraic manipulations are quite easy to automate, including the cancellation of factorials, since we avoid any possible trouble points or singularities.
- There is still currently a bit of manual work involved in showing that suitable limits for end values of G exist.
- However, this could be automated too by arguing that the denominator of the certificate, for fixed n, is a polynomial in k and hence is nonzero for large enough $|k|$.
- We believe this is a satisfying, if somewhat involved, interpretation, and that it justifies the WZ method more clearly.