Cálculo Simbólico

En matemáticas y ciencias de la computación, el cálculo simbólico, también conocido como cálculo algebraico o álgebra computacional, es un área científica que se refiere al estudio y desarrollo de algoritmos y software para la manipulación de expresiones matemáticas y otros objetos matemáticos.

Hablando con propiedad, el álgebra computacional debe ser un sub-campo de la computación científica, pero son considerados generalmente campos distintos, debido a que la computación científica se basa mayormente en el análisis numérico con números aproximados en punto flotante; mientras que, el álgebra computacional enfatiza el cálculo exacto con expresiones que contengan variables que no tienen un valor asociado y son manipuladas como símbolos (de ahí se debe el nombre de cálculo simbólico).

Un sistema computacional de cálculo simbólico es una calculadora avanzada, capaz de realizar operaciones numéricas habituales, más otras de naturaleza algebraica abstracta. Es decir que puede trabajar con fórmulas en las que figuran, indistintamente, números y variables simbólicas no cuantificadas.

Las aplicaciones de software que realizan cálculos simbólicos se conocen como sistemas de álgebra computacional, con el término sistema aludiendo a la complejidad de las principales aplicaciones que incluyen, al menos, un método para representar los datos matemáticos en una computadora, un lenguaje de programación de usuario (por lo general diferente del lenguaje usado para la ejecución), un administrador de memoria, una interfaz de usuario para la entrada/salida de expresiones matemáticas, un gran conjunto de subrutinas para realizar operaciones usuales, como la simplificación de expresiones, la regla de la cadena utilizando diferenciación, factorización de polinomios, integración indefinida, etc.

En los comienzos del álgebra computacional, alrededor de 1970, cuando los algoritmos clásicos se implementaron por primera vez en los ordenadores, resultaron ser altamente ineficientes. Por lo tanto, una gran parte de la labor de los investigadores en el campo consistió en revisar el álgebra clásica con el fin de hacerla más computable y descubrir algoritmos eficientes que implementen esta eficacia.

Este tipo de herramientas se denominan genéricamente CAS (del inglés, computer algebra system). Su objetivo es automatizar la realización cálculos tediosos. Aunque estas herramientas pueden tener diferentes prestaciones, un CAS genérico suele tener capacidades para realizar

- simplificaciones y desarrollos de fórmulas que contienen variables,
- cálculo matricial, derivadas e integrales indefinidas,
- resolver ecuaciones lineales y algunas no lineales de forma exacta,
- resolver ecuaciones de forma numérica
- resolver algunas ecuaciones diferenciales, etc.

También se considera que un CAS de este tipo debe tener facilidades para dibujar gráficos de funciones y poner a disposición del usuario un lenguaje de programación que le permita definir sus propios procedimientos de cálculo.

El álgebra computacional es ampliamente utilizada para experimentar en matemática y diseñar las fórmulas que se utilizan en los programas numéricos. También se usa para cálculos científicos completos, cuando los métodos puramente numéricos fallan, como en la criptografía asimétrica o para algunos problemas no lineales.

Los CAS están influyendo de forma cada vez más significativa en la enseñanza de las matemáticas, especialmente a nivel universitario, por sus posibilidades de apoyo docente en ilustración de ciertos fenómenos y conceptos, en la validación experimental de algunas hipótesis, en el ahorro de tiempo en la realización de tareas conceptualmente simples pero tediosas, etc.

Como el software numérico es muy eficiente para el cálculo numérico aproximado, en álgebra computacional es común enfatizar el cálculo exacto con datos representados de manera exacta.

Una representación exacta implica que, incluso cuando el tamaño del resultado es pequeño, los datos intermedios generados durante el cálculo pueden crecer de forma impredecible. Este comportamiento se llama aumento de expresión. Para obviar este problema, se utilizan diversos métodos en la representación de los datos, así como en los algoritmos que los manipulan.

Números

Los sistemas numéricos usados habitualmente en el cálculo numérico se basan números representados en punto flotante y números enteros con un tamaño fijo acotado. Ninguno de ellos es conveniente para el álgebra computacional, debido al aumento de expresión.

Por lo tanto, los números básicos utilizados en el álgebra computacional son los números enteros, comúnmente representados por una secuencia ilimitada de dígitos con signo en alguna base de numeración, generalmente la base más grande permitida por la palabra de la computadora. Estos enteros permiten definir números racionales, que son fracciones irreducibles de dos enteros.

Programar una implementación eficiente de operaciones aritméticas con números representados de esta manera es una tarea difícil. Por lo tanto, la mayoría de los sistemas informáticos de álgebra gratuitos y algunos comerciales como Mathematica y Maple (software) utilizan la biblioteca GMP, que es un estándar de facto.

Expresiones

A excepción de los números y las variables, toda expresión matemática puede verse como el símbolo de un operador seguido de una secuencia de operandos. En los programas informáticos de álgebra, las expresiones suelen representarse de esta forma. Esta representación es muy flexible, y muchas cosas que a primera vista parecen no ser expresiones matemáticas, pueden ser representadas y manipuladas como tales.

Por ejemplo, una ecuación es una expresión con "=" como operador, una matriz puede representarse como una expresión con "matriz" como operador y sus filas como operandos.

Incluso los programas pueden ser considerados y representados como expresiones con el operador "procedimiento" y, al menos, dos operandos, la lista de parámetros y el cuerpo, que es en sí mismo una expresión con "cuerpo" como operador y una secuencia de instrucciones como operandos.

En sentido inverso, cualquier expresión matemática puede verse como un programa. Por ejemplo, la expresión a + b puede verse como un programa para la suma, con a y b como parámetros. La ejecución de este programa consiste en evaluar la expresión para valores dados de a y b; si no se les da ningún valor, el resultado de la evaluación es simplemente su entrada.

Este proceso de evaluación diferida es fundamental en el álgebra computacional. Por ejemplo, el operador "=" de las ecuaciones es, en la mayoría de los sistemas de álgebra computacional, el nombre del programa para verificar la igualdad. Normalmente, la evaluación de una ecuación da como resultado una ecuación, pero, cuando se necesita verificar la igualdad, ya sea explícitamente a través de un comando de "evaluación booleana" solicitado por el usuario, o iniciado automáticamente por el sistema en el caso de una prueba dentro de un programa, en ese caso, se ejecuta la evaluación obteniendo un resultado booleano.

Como el tamaño de los operandos de una expresión es impredecible y puede cambiar durante una sesión de trabajo, la secuencia de los operandos generalmente se representa como una secuencia de punteros (como en Macsyma) o entradas en una tabla hash (como en Maple).

Simplificación

La aplicación literal de las reglas básicas de derivación de la expresión a^x con respecto a x da como resultado:

$$x \cdot a^{x-1} \cdot 0 + a^x \cdot \left(1 \cdot \log a + x \cdot \frac{0}{a}\right).$$

Dicha expresión resulta muy compicada en la práctica y se desea una expresión más simple, lo cual requiere aplicar simplificación cuando se trabaja con expresiones generales.

Esta simplificación normalmente se realiza mediante reglas de reescritura. Hay varias clases de reglas de reescritura a considerar. Las más simples son reglas que siempre reducen el tamaño de la expresión, como $E - E \to 0$ o $sin(0) \to 0$, las que se aplican sistemáticamente en los sistemas de álgebra computacional.

Las operaciones asociativas como la suma y la multiplicación presentan una dificultad. La forma estándar de tratar con la asociatividad es considerar que la suma y la multiplicación tienen un número arbitrario de operandos, es decir, que a + b + c se representa como "+" (a, b, c). Por lo tanto, a + (b + c) y (a + b) + c se simplifican a "+" (a, b, c), que se muestra como a + b + c.

En el caso de expresiones como a − b + c, la forma más sencilla es reescribir sistemáticamente −E, E − F, E/F como, respectivamente, (−1)·E, E + (−1)·F, E·F−1. En otras palabras, en la representación interna de las expresiones, no hay ni resta ni división ni menos unario, fuera de la representación de los números.

La conmutatividad de la suma y la multiplicación también presentan dificultades. El problema es reconocer rápidamente los términos semejantes para combinarlos o cancelarlos. Probar cada par de términos es costoso con sumas y productos muy largos.

Para abordar esto, Macsyma ordena los operandos de sumas y productos en un orden que coloca términos similares en lugares consecutivos, lo que permite una fácil detección. En Maple, una función hash está diseñada para generar colisiones cuando se ingresan términos similares, lo que permite combinarlos tan pronto como se introducen. Esto permite que las subexpresiones que aparecen varias veces en un cálculo se reconozcan inmediatamente y se almacenen solo una vez. Esto ahorra memoria y acelera el cálculo al evitar la repetición de las mismas operaciones en expresiones idénticas.

Las reglas de reescritura a veces aumentan y otras veces reducen el tamaño de las expresiones a las que se aplican. Este es el caso de la distributividad o identidades trigonométricas.

Por ejemplo, la ley de distributividad permite reescribir

$$(x+1)^4 \to x^4 + 4x^3 + 6x^2 + 4x + 1$$

y

$$(x-1)(x^4 + x^3 + x^2 + x + 1) \to x^5 - 1.$$

Como no hay forma de hacer una buena elección de aplicar (o no) dicha regla de reescritura en general, la reescritura se realiza solo cuando el usuario la invoca explícitamente.

Para la distributividad, la función que aplica esta regla de reescritura generalmente se denomina "expandir". La regla de reescritura inversa, llamada "factor", requiere un algoritmo no trivial, que es, por lo tanto, una función clave en los sistemas de álgebra computacional.

Aspectos matemáticos

Algunas preguntas fundamentales surgen cuando uno quiere manipular expresiones matemáticas en una computadora. Consideramos principalmente el caso de las fracciones racionales con varias variables. Esta no es una restricción real, ya que, en cuanto se simplifican las funciones irracionales que aparecen en una expresión, se suelen considerar como nuevos indeterminados. Por ejemplo, $(\sin(x+y)^2 + \log(z^2-5))^3$ se puede ver como un polinomio en $\sin(x+y)$ y $\log(z^2-5)$.

Igualdad

Hay dos nociones de igualdad de expresiones matemáticas. La igualdad sintáctica es la igualdad de las expresiones que significa que se escriben (o representan en una computadora) de la misma manera. Como es trivial, es raramente considerado por los matemáticos, pero es la única igualdad que es fácil de probar con un programa. La igualdad semántica es cuando dos expresiones representan el mismo objeto matemático , como en $(x+y)^2 = x^2 + 2xy + y^2$,

Se sabe que no existe un algoritmo que permita decidir si dos expresiones que representan valores numéricos son semánticamente iguales, si se permiten exponentes y logaritmos en las expresiones. Por lo tanto, (semánticamente) la igualdad sólo se puede probar en algunas clases de expresiones tales como los polinomios y las fracciones racionales.

Para probar la igualdad de dos expresiones, en lugar de diseñar un algoritmo específico, es habitual ponerlos en alguna forma canónica o poner su diferencia en una forma normal y probar la igualdad sintáctica del resultado.

En álgebra computacional, "forma canónica" y "forma normal" no son sinónimos.

- Una forma canónica es tal que dos expresiones en forma canónica son semánticamente iguales si y solo si son sintácticamente iguales, mientras que
- una forma normal es tal que una expresión en forma normal es semánticamente cero solo si es sintácticamente cero.

En otras palabras, el cero tiene una representación única como expresión en forma normal.

Las formas normales son generalmente preferidas en álgebra computacional por varias razones. En primer lugar, las formas canónicas pueden ser más costosa de calcular que las formas normales. Por ejemplo, para poner un polinomio en forma canónica, se tiene que ampliar por distributividad cada producto, si bien no es necesario, con una forma normal.

En segundo lugar, puede darse el caso, como en el caso de las expresiones que implican radicales, de que una forma canónica, si existe, dependa de algunas elecciones arbitrarias y que estas elecciones puedan

Maxima

Las nuevas orientaciones impulsadas por el Espacio Europeo de Educación Superior aconsejan la introducción de instrumentos y recursos que faciliten el aprendizaje autónomo-dirigido de los estudiantes. Y los recursos informáticos de tipo CAS pueden prestar un ayuda significativa a los estudiantes para cálculo simbólico o numérico, en el estudio de funciones y sus propiedades apoyado por la representación gráfica, y en la adquisición e interiorización, a través de la experimentación con los elementos anteriores, de las ideas que sustentan el conocimiento abstracto en matemáticas.

Los CAS más conocidos, como ©Derive, ©Maple, ©Mathematica..., son productos comerciales de precios elevados. Maxima no sólo es gratuito, es un programa de código abierto, cuyo código fuente está disponible bajo licencia GPL, pudiendo ser compilado en cualquier sistema operativo y distribuido libremente bajo los términos de dicha licencia.

Maxima desciende de Macsyma, el legendario sistema de álgebra computacional desarrollado a finales de 1960 en el Instituto Tecnológico de Massachusetts (MIT) como programa propietario, anterior a los CAS antes señalados.

Bibliografía

- Cálculo simbólico (https://es.wikipedia.org/wiki/C%C3%A1lculo_simb%C3%B3lico)
- Computer algebra (https://en.wikipedia.org/wiki/Computer_algebra)
- Cálculo Simbólico (https://www.um.es/innova/OCW/informatica-para-universitarios/ipu_docs/calculo_simbolico/ipu_calculo_simbolico.html)