Arithmetic in ACL2

Guillermo Cabrera

Introduction

- Formalizing sums and properties
  - Notation

$$\sum_{k=1}^{n} a_k \qquad \sum_{1 \le k \le n} a_k$$

- Machine learning as motivation

$$\mu_j := \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$$

Manipulation of sums

- Distributive law

$$\sum_{k\in K} ca_k = c\sum_{k\in K} a_k\,;$$

- Associative law

$$\sum_{k\in K}(a_k + b_k) = \sum_{k\in K} a_k + \sum_{k\in K} b_k$$

- Commutative law

$$\sum_{k\in K} a_k = \sum_{p(k)\in K} a_{p(k)}\,.$$

Types of Sums

- Finite terms

$$\sum_{i=1}^{n} i^2 = (n(n+1)(2n+1)) / 6$$

- Infinite terms
  - Goldbach - Euler theorem

$$\sum_{k \in P} 1 / (k-1) = 1/3+1/7+1/8+1/15 + \ldots$$

$$= 1$$

Example

```
(defun sum1128 (n)
  (if (zp n)
      0
    (+ (* n n) (sum3 (1- n)))))

(defthm sum1128-thm
  (implies (natp n)
       (equal (sum3 n)
                 (/ (* n (+ n 1) (+ (* 2 n) 1)) 6))))
```

- For `BINARY-+` we start with:
  - (:REWRITE DEFAULT-+-2) → (+ X Y) = (FIX X)
  - (:REWRITE INVERSE-OF-+) → (+ X (- X)) = 0
  - (:REWRITE UNICITY-OF-0) → (+ 0 X) = (FIX X)
  - (:REWRITE COMMUTATIVITY-OF-+) → (+ X Y) = (+ Y X)
  - (:REWRITE ASSOCIATIVITY-OF-+) → (+ (+ X Y) Z) = (+ X Y Z)

Example

```
Subgoal *1/4'
(IMPLIES (AND (NOT (ZP N))
              (EQUAL (SUM3 (+ -1 N))
                     (+ -1/6 (* 1/6 N)
                        1/6 (* -1/6 N)
                        (* (+ -1 N) 1/6 N)
                        1/3 (* -1/3 N)
                        -1/3 (* 1/3 N)
                        (* (+ -1 N) -1/3 N)
                        (* (+ -1 N) (+ 1 -1 N) 1/6 2 N)))
              (<= 0 N))
         (EQUAL (+ (SUM3 (+ -1 N)) (* N N))
                (+ (* 1/6 N)
                   (* N 1/6 N)
                   (* N (+ 1 N) 1/6 2 N))))
```

Example

```
Subgoal *1/4'
(IMPLIES (AND (NOT (ZP N))
              (EQUAL (SUM3 (+ -1 N))
                     (+ -1/6 (* 1/6 N)
                        1/6 (* -1/6 N)
                        (* (+ -1 N) 1/6 N)
                        1/3 (* -1/3 N)
                        -1/3 (* 1/3 N)
                        (* (+ -1 N) -1/3 N)
                        (* (+ -1 N) (+ 1 -1 N) 1/6 2 N)))
              (<= 0 N))
         (EQUAL (+ (SUM3 (+ -1 N)) (* N N))
                (+ (* 1/6 N)
                   (* N 1/6 N)
                   (* N (+ 1 N) 1/6 2 N))))
```

Example

```
Subgoal *1/4'
(IMPLIES (AND (NOT (ZP N))
              (EQUAL (SUM3 (+ -1 N))
                     (+ -1/6 (* 1/6 N)
                        1/6 (* -1/6 N)
                        (* (+ -1 N) 1/6 N)
                        1/3 (* -1/3 N)
                        -1/3 (* 1/3 N)
                        (* (+ -1 N) -1/3 N)
                        (* (+ -1 N) (+ 1 -1 N) 1/6 2 N)))
              (<= 0 N))
         (EQUAL (+ (SUM3 (+ -1 N)) (* N N))
                (+ (* 1/6 N)
                   (* N 1/6 N)
                   (* N (+ 1 N) 1/6 2 N))))
```

- Additions
  - (:REWRITE COMMUTATIVITY-2-OF-*) → (* Y X Z) = (* X Y Z)
  - (:REWRITE FOLD-CONST-IN-*) → (* X Y Z) = (* (* X Y) Z)

Proved Properties

- Alternate forms for sums

```
(/ (* n (1+ n) (+(* 2 n) 1)) 6)
= (* n (+ (* (+ (/ n 3) (/ 1 2)) n)(/ 1 6)))
= (+ (/ (* n n n) 3) (/ (* n n) 2) (/ n 6))
= (+ (/ (* 2 n n n) 6) (/ (* 3 n n) 6) (/ n 6))
```

- Associative and distributive for instance of a sum

```
(defun sumLhs (k)
  (if (zp k)
    0
    (+ k k (sumLhs (- k 1)) )))
```

```
(defun sumRhs (k)
  (if (zp k)
    0
    (+ k (sumRhs (- k 1)))))
```

```
(defthm associativeLawSums
  (implies (natp k)
         (equal (sumLhs k) (+ (sumRhs k) (sumRhs k)))))
```

Problems Encountered

- Commutative
  - How to represent a permutation
  - Bernoulli trail with RANDOM$?
  - Generate all possible combinations for K?

$$\sum_{k \in K} a_k = \sum_{p(k) \in K} a_{p(k)} .$$

- Representing infinite sums
  - How to handle infinity

Conclusion

- Alternate forms for formulas gave insight to needed lemmas
- Subtle differences when compared to list data type
- Developing model for finite term sums
  - Prove sum properties on this representation
  - Example in [Gamboa2001] with sumlists