Optimal Addition Chains and the Scholz Conjecture

Zara Siddique

December 8, 2015

1 Introduction

An addition chain can be described as sequence of natural numbers (starting at 1), such that each element is the sum of two numbers previously in the sequence.

This can be written mathematically as:

$$v = (v_0, ..., v_s), \text{ with } v_0 = 1 \text{ and } v_s = n$$

$$\text{for each } v_i \text{ holds: } v_i = v_j + v_k \text{ where } 0 \leq j, k \leq i - 1 \text{ [1]}$$

An optimal addition chain refers to the addition chain of shortest length for n, denoted l(n). For example, l(5) = 3 (since the shortest chain is 1 + 1 = 2, 2 + 2 = 4, 4 + 1 = 5). [2] A key application of this is addition-chain exponentiation, where the optimal addition chain can be used to calculate positive integer powers with a minimal number of multiplications.

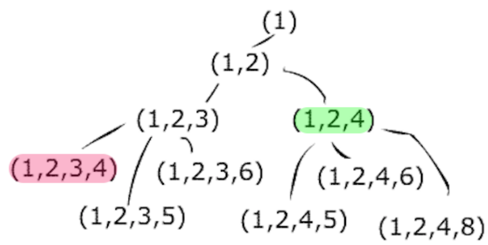

Figure 1: The first 10 addition chains: for both the highlighted addition chains, n = 4, however there exists an optimal chain [3].

2 Finding the length of the optimal addition chain

The difficulty in finding optimal addition chains is that there is no way to calculate one directly (yet). For example, consider the following addition chains for n = 15:

$$l(15) = (1, 2, 4, 8, 12, 13, 15) = 6$$

$$l(15) = (1, 2, 3, 6, 12, 15) = 5$$

While it intuitively makes sense to calculate the largest number possible to get closer to n, this example shows that early on in the addition chain, small unpredictable changes can affect the final length. To get around this problem, the code below calculates every possible addition chain, and (if n is found,) compares the length of all addition chains containing n to find the one of shortest length.

2.1 Using Python

The following code uses three functions to return a list of the lengths of optimal addition chains (to save space I have omitted some of the code and comments - additional comments can be found on the complete file at `https://cloud.sagemath.com/projects/a54c7544-f743-457c-af0a-ab38ce1ebbd5/files/Addition%20Chains%20and%20Scholz%20Conjecture.py`).

```
pairlist = []
megalist = [[1]]

def mainfunc(number):
    """
    Finds the length of the optimal addition chain of any integer
    """
    q = 0
    for i in range(q, len(megalist)):
     addaterm(megalist[i])
    q = len(megalist)
    if checker(number):
        del megalist[:]
        megalist.append([1])
    else:
        mainfunc(number)

def addaterm(path):
    """
    Computes all possible addition chains and appends them to megalist
    """
    for i in range(0, len(path)):
        newlist = path
        newlist.append(path[i] + path[-1])   # (Brauer Chain)
        megalist.append(newlist[0:len(newlist)])
        newlist.pop()

def checker(number):
    """
    Checks megalist for the number as the last element of a list, and if found, will
    find the length of the optimal addition chain and append it to pairlist
    """
    lengths = []
    for i in range(0, len(megalist)):
        if megalist[i][-1] == number:
            lengths.append(len(megalist[i]) - 1)
            pairlist.append([number, min(lengths)])
            return True
```

3 The Scholz Conjecture

The most famous unsolved problem related to addition chains is Scholz's conjecture, first noted by Arnold Scholz in 1937, and Alfred T. Brauer in 1939. It states that for all natural numbers n:

$$I(2^n - 1) \leq n - 1 + I(n)$$

While this conjecture has been proven for certain classes of n (such as Hansen and Brauer numbers [4]), its validity for all natural numbers remains an open problem.

The code below follows on from that above, and will return whether the Scholz conjecture is true for any natural number. In the full code linked previously, it is shown to be true for numbers 1 to 8.

```
def scholz(number):
    a = number
    mainfunc(number)
    for i in range(0, len(pairlist)): # Sets b to l(a)
```

```
        if pairlist[i][0] == number:
            b = pairlist[i][1]
    c = (2 ** number) - 1
    mainfunc(c)
    for i in range(0, len(pairlist)): # Sets d to l(c)
        if pairlist[i][0] == c:
            d = pairlist[i][1]
    if d <= a - 1 + b:  # Scholz Conjecture
        return True
```

The main problem with computing the Scholz conjecture is the large memory usage and slow timing, particularly with larger numbers; so far the Scholz conjecture has been proven true (and checked by computer) for all natural numbers up to 5784689 (the first non-Hansen number) [1].

4 Conclusion

Calculating optimal addition chains for large numbers is possible but requires large memory usage and a lot of time. The next step in investigating addition chains would be to find the most efficient way of calculating an optimal addition chain or its length, something that hasn't yet been done. Considerable speed ups can be gained by exploiting overlapping calculations and doing them simultaneously [5].

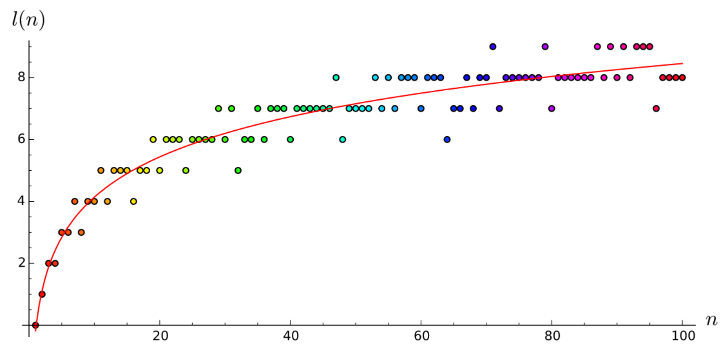

Figure 2: Scatter graph of n and l(n) and best fit line - could this be used to estimate l(n)? Full code used available at `https://cloud.sagemath.com/projects/a54c7544-f743-457c-af0a-ab38ce1ebbd5/files/Graph.sagews`

Another area to explore is the mathematical side of Scholz's Conjecture, which has not yet been proven for non-Hansen numbers. A method in which to predict optimal addition chains or their length could also be investigated (see Fig. 2). Similar conjectures such as $l(2n) \geq l(n)$ (disproven by Clift in 2010) also contain a large scope for exploration [5].

References

[1] Wikipedia, "Addition chain — wikipedia, the free encyclopedia," 2015. [Online; accessed 7-December-2015].

[2] Wikipedia, "Scholz conjecture — wikipedia, the free encyclopedia," 2014. [Online; accessed 7-December-2015].

[3] W.-L. D. Tseng, "Addition chains," 2008. [Online; accessed 7-December-2015].

[4] R. Guy, *Unsolved Problems in Number Theory*. Problem Books in Mathematics, Springer New York, 2013.

[5] N. M. Clift, "Calculating optimal addition chains," 2010. [Online; accessed 7-December-2015].