Computing (2011) 91:265–284
DOI 10.1007/s00607-010-0118-8

Calculating optimal addition chains

Neill Michael Clift

Received: 16 January 2010 / Accepted: 22 August 2010 / Published online: 11 September 2010

Abstract An addition chain is a finite sequence of positive integers $1 = a_0 \leq a_1 \leq \cdots \leq a_r = n$ with the property that for all $i > 0$ there exists a j, k with $a_i = a_j + a_k$ and $r \geq i > j \geq k \geq 0$. An optimal addition chain is one of shortest possible length r denoted $l(n)$. A new algorithm for calculating optimal addition chains is described. This algorithm is far faster than the best known methods when used to calculate ranges of optimal addition chains. When used for single values the algorithm is slower than the best known methods but does not require the use of tables of pre-computed values. Hence it is suitable for calculating optimal addition chains for point values above currently calculated chain limits. The lengths of all optimal addition chains for $n \leq 2^{32}$ were calculated and the conjecture that $l(2n) \geq l(n)$ was disproved. Exact equality in the Scholz–Brauer conjecture $l(2^n - 1) = l(n) + n - 1$ was confirmed for many new values.

Keywords Addition chains · Exponentiation · Scholz–Brauer · Sequences · Optimization

Mathematics Subject Classification (2000) 11Y16 · 11Y55 · 05C20

1 Introduction

An *addition chain* A is a finite sequence of positive integers called *elements* $1 = a_0 \leq a_1 \leq \cdots \leq a_r = n$ with the property that for all $i > 0$ there exists a j, k

Communicated by C.C. Douglas.

N. M. Clift (✉)
Microsoft Corporation, 1 Microsoft Way, Redmond, WA 98052, USA
e-mail: neillclift@live.com

with $a_i = a_j + a_k$ and $r \geq i > j \geq k \geq 0$. This is called an addition chain of length r for n. We call n the *target* of an addition chain. An *optimal* addition chain has shortest possible length denoted by $l(n)$ and is a strictly increasing sequence as duplicate chain elements could be removed to shorten the chain. It is clear that any addition chain whose elements were not ordered could be simply rearranged to one that is and any elements greater than the target could be deleted. Chains therefore are simply defined as being ordered without any loss of generality.

Interest in optimal addition chains arises naturally in systems where all multiplications have equivalent cost and the evaluation of x^n needs to be as quick as possible for many different values of x with fixed n. The problem appears to have first been considered in a question [1] and its answer [2] with the term addition chain first appearing in [3]. The most complete coverage of the subject area appears in [4] and the review paper [5]. All possible optimal addition chains for 7 are given as an example:

$$12347, \quad 12357, \quad 12367, \quad 12457, \quad 12467$$

This illustrates two important points. First that optimal addition chains are not necessarily unique and second that elements of an addition chain may be formed in more than one way ($4 = 3 + 1$ or $4 = 2 + 2$ in 1 2 3 4 7).

Exponentiation has become important as many cryptographic algorithms have this operation at their core [6,7]. Optimizing exponentiation can have significant impact on algorithm run time and many near optimal methods are known [8]. Tables of optimal addition chain lengths have been used to benchmark new algorithms [9]. Many counterintuitive properties of $l(n)$ are known and this makes the subject interesting to study.

This paper is organized as follows. We will begin in Sect. 2 with some basic definitions commonly used in the literature of addition chains. Then in Sect. 3 we will outline the graphical representation of addition chains and cover the standard graph reduction in Sect. 4 that will remove some redundancy in the numerical representation. In Sect. 5 we will prove that certain graph structures cannot be present in optimal chains as well as structures that must be present in at least one chain for a particular target in Sect. 6. These graph techniques will have a large impact on the runtime of searching for optimal chains. We will develop in Sect. 7 a simple relationship between the vertex degree of a graph and the complexity of its associated addition chain and this will enable limiting the search space and naturally lead to a simple and very effective heuristic for search order. In Sect. 8 we will prove that in some sense deleting addition chain elements can only make chains more efficient and this will enable us to prune early during a search based on what we know the chain must contain later. In Sect. 9 we show a simple graph rearrangement that reduces the search space still further. In Sect. 10 we describe the basic graph enumeration technique used to find optimal addition chains and introduce a dynamic bounding technique that gradually becomes more restrictive as solutions are found. In Sect. 11 we will show how to condense knowledge of later chain construction (driven by required graph structures) into only two quantities and show how this can be used to prune very early in the search space. All of the ideas will come together in the pseudo-code of the algorithm in Sect. 12. In Sect. 13 we will outline the results of running the new algorithm. We will find counter-examples to some longstanding conjectures, extend some conjectures to new, much larger limits, exclude the potential of proving the Scholz–Brauer conjecture via one particular chain type attack and extend the table of some basic functions. Finally we will cover some future research directions in Sect. 14.

2 Basic definitions

The standard notations of [4] where they exist will be used. The construction of each element of an addition chain is called a *step*. For an addition chain $1 = a_0 \leq a_1 \leq \cdots \leq a_r = n$ different step types are classified as follows:

Doubling step: $a_i = 2a_{i-1}, i > 0$

Non-doubling step: $a_i = a_j + a_k, i > j > k \geq 0$

Notice that steps of the form $a_i = 2a_j$, $j \leq i-2$ are defined as non-doubling steps.

The following useful functions are defined: $\lambda(n) = \lfloor \log_2 n \rfloor (n \geq 1)$ and $v(n)(n \geq 1)$ the number of 1 bits in the binary representation of n:

$$v(1) = 1, \quad v(2n) = v(n), \quad v(2n+1) = v(n) + 1$$

This definition along with the monotonically increasing definition of the chain and the fact that $a_i \leq 2a_{i-1}(i > 0)$, gives only two possible step types classified as:

Big step: $\lambda(a_i) = \lambda(a_{i-1}) + 1$

Small step: $\lambda(a_i) = \lambda(a_{i-1})$

Doubling steps (or doublings) are always big steps but not all big steps are doublings. This leads naturally [10] to splitting $l(n)$ into two components:

$$l(n) = \lambda(n) + S(n).$$

The variable portion $S(n) \geq 0$ is defined as the number of small steps in an optimal addition chain for n. Because $\lambda(n)$ is fixed for a given positive integer, finding optimal addition chains amounts to minimizing the number of small steps across all possible chains. Hansen [10] remarked that $S(n)$ could be regarded as a measure of the difficulty or complexity of a number n and that for each element $a_i(0 \leq i \leq r)$ in an optimal addition chain for n we must have $S(a_i) \leq S(n)$. This notion of complexity is borne out by computer calculations of optimal addition chains where the dominant factor in runtime is the small step counts [11, pp 1261–1262].

Given an addition chain A with elements $a_i(0 \leq i \leq r)$ it is useful to refer to the number of small steps in the chain up to and including a_i. This is denoted by:

$$S_A(a_i) = i - \lambda(a_i), \quad (i \geq 0).$$

Likewise the number of small steps between two elements $a_i, a_j(i \geq j \geq 0)$ is denoted with:

$$S_A(a_i, a_j) = i - j - (\lambda(a_i) - \lambda(a_j)), \quad (i \geq j \geq 0).$$

Table 1 The two formal chain mappings for the chain 1 2 3 4 7

i	0	1	2	3	4	i	0	1	2	3	4
a_i	1	$2=1+1$	$3=2+1$	$4=3+1$	$7=4+3$	a_i	1	$2=1+1$	$3=2+1$	$4=2+2$	$7=4+3$
$\gamma(i)$		0	1	2	3	$\gamma(i)$		0	1	1	3
$a_{\gamma(i)}$		1	2	3	4	$a_{\gamma(i)}$		1	2	2	4
$\delta(i)$		0	0	0	2	$\delta(i)$		0	0	1	2
$a_{\delta(i)}$		1	1	1	3	$a_{\delta(i)}$		1	1	2	3

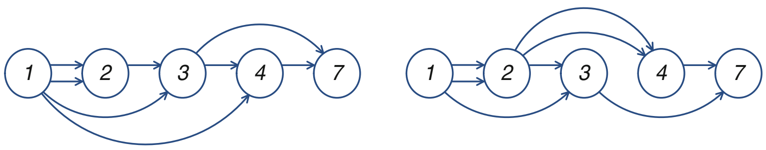

Fig. 1 The two graphical representations of the optimal addition chain 1 2 3 4 7

Sometimes there is a need to resolve the ambiguity in exactly how a term in an addition chain is formed. For example if $a_i = a_j + a_k$ then two mappings $\gamma : [1, r] \mapsto [0, r-1], i \xrightarrow{\gamma} j, \delta : [1, r] \mapsto [0, r-1], i \xrightarrow{\delta} k$ are defined that describe how the sum was formed. An addition chain along with these mappings will be called a *formal* addition chain. Table 1 shows two formal addition chains for 1,2,3,4,7 differing in the choice made for the construction of $a_3 = 4$.

3 Graphical representation

A multi-digraph G is a finite non-empty set of objects called vertices denoted by V with a multi-set of ordered vertex pairs called arcs denoted by E. A multi-set is a set where duplicate elements are allowed. We say that an arc goes from a vertex $u \in V$ to a vertex $v \in V$ if $(u, v) \in E$.

We may represent a formal addition chain A of length r as an acyclic multi-digraph by taking each chain element as a vertex with arcs from the two elements used to form it via addition (except the element 1 that is the starting point of the chain) [4, pp 480–481]. Formally for an addition chain A of length r we have a multi-digraph $G_A = (V, E, \alpha, \omega)$ where V is the set of vertices, E the set of arcs and α, ω are mappings that take an arc to its start and end vertex respectively. This gives $V = \{v_i : 0 \le i \le r\}$ and $E = \{(v_{\gamma(i)}, v_i), (v_{\delta(i)}, v_i) : 1 \le i \le r\}$, $\alpha, \omega : E \mapsto V, (v_1, v_2) \xrightarrow{\alpha} v_1, (v_1, v_2) \xrightarrow{\omega} v_2$. We label each vertex with its numerical value from the addition chain. The graphical representation of an addition chain appeared in [12] derived from a usage in [13].

See Fig. 1 for an example where the optimal addition chain 1, 2, 3, 4, 7 is represented graphically as two graphs because of the ambiguity in the construction of the value 4 (1+3 or 2+2).

The vertex and arc sets for the first graph in Fig. 1 using the vertex labels for clarity are $V = \{1, 2, 3, 4, 7\}$ and $E = \{(1, 2), (1, 2), (1, 3), (1, 4), (2, 3), (3, 4), (3, 7), (4, 7)\}$. This clearly shows the multi-set nature of the arc set due to the two arcs between vertex 1 and 2.

We define the *in-degree* of a vertex $d^+(v) = |\{e \in E : \omega(e) = v\}|$ and the *out-degree* in a similar way $d^-(v) = |\{e \in E : \alpha(e) = v\}|$. Each graph will have an in-degree of 2 for each vertex except for the single vertex labeled 1. This we call the *source* of the graph or source vertex. In a graph for an optimal addition chain each vertex except for the single vertex labeled with the target has an out-degree of at least 1. If a vertex other than the target had an out-degree of 0 then this vertex and its associated edges could be deleted leading to a shorter chain. The vertex would have represented an element calculated in the chain that was not used in the construction of the target. We call the vertex associated with the target with 0 out-degree the *sink* vertex. We define the in-degree from vertex u to vertex v by $d^+(u, v) = |\{e \in E : \alpha(e) = u, \omega(e) = v\}|$. If we have multiple arcs between two vertices ($d^+(u, v) > 1$) then we describe these as *parallel* arcs.

3.1 Path counts and addition chain lengths

For an addition chain A with associated graph $G = (V, E, \alpha, \omega)$, we define a path count operator that gives the total number of paths from the source vertex ($v_0 \in V$) to the given vertex:

$$p(v_i) = \begin{cases} 1, & \text{if } i = 0 \\ \sum_{e \in E, \omega(e) = v_i} p(\alpha(e)), & \text{if } i > 0 \end{cases}$$

We have the relationship that $p(v_i) = a_i$ and hence our vertex labels correspond to path counts. We can prove this with induction on $v_i (0 \le i \le r)$, the vertices derived from the addition chain elements $a_i (0 \le i \le r)$. By definition $p(v_0) = 1 = a_0$ and we assume the relationship is true for $0 \le i \le r$. We show that the property holds for $i + 1$ and hence the relationship always holds:

$$p(v_{i+1}) = \sum_{e \in E, \omega(e) = v_{i+1}} p(\alpha(e)) = \sum_{j \in \{\gamma(i+1), \delta(i+1)\}} a_j = a_{i+1}$$

Given an arbitrary acyclic graph $G = (V, E, \alpha, \omega)$ with a single source $v_0 \in V$ and $d^+(v) = 2, \forall v \in V \backslash v_0$ we can construct an *associated* addition chain A. We do this without loss of generality by labeling the vertices such that $p(v_0) \le p(v_1) \le \cdots \le p(v_{|V|-1})$ and form $a_i = p(v_i)$. This vertex ordering will be used often and we will call it *path count order*.

We can obtain the length r of an associated addition chain for a graph $G = (V, E, \alpha, \omega)$ from:

$$r = |E| - |V| + 1 \text{ since } |E| = 2(|V| - 1) \quad \text{and} \quad |V| = r + 1.$$

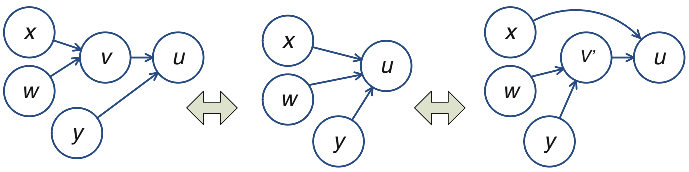

Fig. 2 Reduction and expansion of a graph with a vertex with an out-degree of one

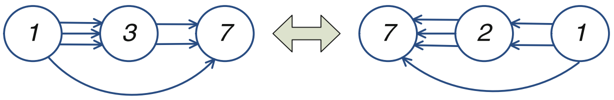

Fig. 3 Conversion of a graph to its dual

This holds because there are two arcs that enter each vertex apart from the source. We use the more complex formula rather than $|V| - 1$ as this will be invariant under later rearrangements where we delete vertices and arcs.

4 Graph reduction

The sequence form of an addition chain often hides a redundancy in the representation that becomes clear in the graphical representation [4, p 480]. Let A be an optimal addition chain of length r with associated graph $G = (V, E, \alpha, \omega)$ and $v_0, \ldots, v_{|V|-1} \in V$ in path count order. If we have a vertex $v \in V$ with $d^-(v) = 1$ we find that the arc $e \in E$ with $\alpha(e) = v$ leads to a vertex u with $\omega(e) = u$ and hence u is formed by the sum of paths from 3 vertices that can be taken in any order. We can make this more explicit by deleting the vertex v and arc e and changing all arcs that terminate at v to instead terminate at u. This process called *reduction* is reversible (called *expansion*) and allows the transformation of one chain into another by taking the vertices in different orders when expanding. This transformation is shown in Fig. 2. This reduction can be applied iteratively until $\forall v \in V \backslash \{v_{|V|-1}\}, d^-(v) \geq 2$. The resulting graph G' is called the *reduced* graph of G associated with addition chain A. Reduction preserves the property that $r = |E| - |V| + 1$ since each reduction reduces both the number of vertices and arcs by 1. Path counts are also preserved for the remaining vertices.

An interesting property of reduced graphs of optimal addition chains is that we may form the *dual* G' of a reduced graph G by reversing all the arcs [12]. The source and sink reverse roles but the path count remains the same for the target. Hence the dual is a potentially different optimal addition chain for the same target. Fig. 3 shows the reduced graph for the addition chain 1 2 3 6 7 and its dual to which a different addition chain 1 2 3 5 7 is associated.

Let us now by way of illustration manually expand the reduced graph of the dual in Fig. 3 to obtain all the addition chains it is associated with. Vertex 1 by definition represents the addition chain element 1. Vertex 2 is formed by the sum of only two values and that can be performed in only 1 way. This gives us $A = 1, 2, \ldots$. Vertex 7 is formed by the sum of 4 values and there are 2 different ways of starting (1+2 or

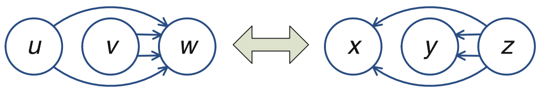

Fig. 4 Multiple sets of parallel arcs in the construction of a vertex and its dual

2+2). So we obtain $A = \begin{cases} 1, 2, 3, \dots \\ 1, 2, 4, \dots \end{cases}$ In the first case we have 2 arcs from 2 and one from 3 used to construct 7 and so we have two ways to proceed ($2 + 2$ or $3 + 2$). In the second case we have 1 arc from each of 1,2 and 4 used to construct 7 and so we have 3 different ways of proceeding ($4 + 2$, $4 + 1$ or $2 + 1$).

This yields $A = \begin{cases} 1, 2, 3, 4, 7 \\ 1, 2, 3, 5, 7 \\ 1, 2, 4, 6, 7 \\ 1, 2, 4, 5, 7 \end{cases}$ after sorting and removing duplicates.

In [14] it was proved that graph reduction partitions the set of formal addition chains for a specific target into equivalence classes.

5 Impossible graph structures

We will now describe some limits to parallel arcs in optimal chains. Firstly we may have no more than 3 parallel arcs:

Theorem A (Flammenkamp 1999) *For an optimal addition chain A with associated reduced graph $G = (V, E, \alpha, \omega)$ we must have $\forall u, v \in V, d^+(u, v) \leq 3$. This appears as Lemma 2 in* [15].

Secondly we can only have parallel arcs from at most one vertex in the construction of another vertex. See the left graph fragment in Fig. 4. By looking at the dual we can deduce that each vertex can only be used at most once with parallel arcs. See Fig. 4 for the transformation.

Theorem B *For an optimal addition chain A with associated reduced graph $G = (V, E, \alpha, \omega)$ we must have $\forall u, v, w \in V, u \neq v, w \neq v, w \neq u, d^+(u, w) \leq 1$ or $d^+(v, w) \leq 1$.*

Proof We assume contrary to the Theorem that we have a vertex pair $u, v \in V$ both used with parallel arcs in the construction of the vertex $w \in V$. We can expand a portion of the chain numerically to give $p(u), 2p(u), 2p(u) + p(v), 2p(u) + 2p(v), \dots,$. We could however produce a shorter chain than this by forming $p(u),\ p(u)+p(v),\ 2p(u)+2p(v), \dots,$ so the chain could not have been optimal which is a contradiction. □

Corollary *By considering the dual $G' = (V', E', \alpha', \omega')$ of the reduced graph G in Theorem B we may deduce that $\forall u, v, w \in V',\ v \neq w, d^+(u, v) \leq 1$ or $d^+(u, w) \leq 1$.*

6 Required graph structures

It is clear from Theorem A that any reduced graph for an optimal formal addition chain must start in one of three ways. We label the vertices in path count order as $v_0, \dots, v_{|V|-1}$:

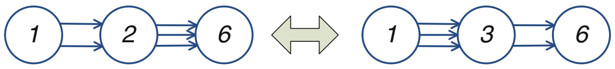

Fig. 5 Moving a doubling from the start to the end of a graph

1) $p(v_0) = 1, |V| = 1$
2) $p(v_0) = 1, p(v_1) = 2, |V| > 1$
3) $p(v_0) = 1, p(v_1) = 3, |V| > 1$

This arises simply because of the imposed limits on the number of arcs in an optimal formal chain's associated reduced graph. Case 1 is the special addition chain for 1 that has no arcs.

For numbers requiring small steps in their addition chains there is always an addition chain that uses the value 1 at least three times leading to at least two odd values in the chain. To prove this we will use the transformation depicted in Fig. 5.

Theorem C *There must exist in the set of all optimal addition chains for n with* $S(n) \geq 1$ *at least one formal addition chain A and its associated reduced graph* $G = (V, E, \alpha, \omega)$ *in path count order with* $d^-(v_0) \geq 3, v_0 \in V$.

Proof Assume that A is an arbitrary optimal addition chain for an arbitrary n with $S(n) \geq 1$. In the associated reduced graph if $d^-(v_0) > 2$ then this addition chain satisfies the Theorem so we may assume that $d^-(v_0) \leq 2$. If $d^-(v_0) = 0$ then this must be the addition chain for $n = 1$ but that contradicts $S(n) \geq 1$. $d^-(v_0) = 1$ would have been eliminated in the graph reduction process leaving the only possibility as $d^-(v_0) = 2$. The chain must start $1, 2, \ldots,$ with no further references to 1 (see the first graph in Fig. 5 as an example). Hence all addition chain elements except the first are even. We can divide each element of the chain apart from the first by 2 and move the first doubling step to the end of the chain so it ends $\ldots, n/2, n$. This process can be iteratively applied to find a graph with the required properties, otherwise it never terminates when the addition chain contains only doubling steps and hence would have $S(n) = 0$. □

Theorem D *If all optimal formal addition chains for n with* $S(n) \geq 2$ *and their associated reduced graphs with vertices in path count order begin as case 3 above* $(p(v_1) = 3)$ *then there must exist at least one optimal formal addition chain A for n and its associated reduced graph* $G = (V, E, \alpha, \omega)$ *in path count order with* $d^-(v_0) \geq 4, v_0 \in V$.

Proof Assume that n is a counter example to the Theorem and select an arbitrary formal optimal addition chain A for n. The associated reduced graph for A is $G = (V, E, \alpha, \omega)$ with the vertices in path count order. We must have $d^-(v_0) \leq 3$ and, in fact $d^-(v_0) = 3$ since $p(v_1) = 3$. The vertex describes the starting value 1 and we must have $n = 3m, m \in \mathbf{N}$. This follows because there must be no further references to 1 after the formation of 3 forcing all elements starting with 3 to be multiples of 3. If $|V| = 2$ then this is the addition chain 1, 2, 3 corresponding to the reduced graph, and this is not possible since the addition chain has one small step but it is given that $S(n) \geq 2$. Hence, $|V| > 2$.

The original addition chain A can be transformed into formal optimal addition chain A' for n by removing the first two elements (1,2) and dividing the remaining elements by 3. This will yield an addition chain for m that can be extended to form an addition chain for n by the addition of the elements $2m, 3m$ to the end. This formal addition chain must be optimal as it is the same length as the original chain A. The associated reduced graph for A' is $G' = (V', E', \alpha', \omega')$ with the vertices $(v'_x \in V', 0 \leq x < |V'|)$ in path count order.

We can now assert that $p(v_2) = 9$ otherwise $p(v'_1) \neq 3$ in A' contrary to the given that $p(v'_1) = 3$ for all addition chains for n. Likewise we must have $d^-(v_1) = 3$ otherwise $d^-(v'_0) > 3$ contrary to n being a counter example to the Theorem.

In summary we must have $p(v_0) = 1$, $p(v_1) = 3$, $p(v_2) = 9$, $d^-(v_1) = 3$, and this chain could be rearranged as 1,2,4,8,9, which is a contradiction for the need to start the reduced graph 1,3. □

Theorems C and D allow us to consider only graphs with the required out-degree for the source vertex and hence reduce the number of cases we need to consider. We will introduce other chain rearrangements but they will not reduce the out-degree of the source vertex.

7 In/out-degree limits

It is natural to ask what are the limits of the in/out degree of a vertex in a reduced graph for an associated optimal addition chain for n. Bounding this will enable enumeration to be faster by limiting the cases needed to be considered and will also hint at a heuristic for searching the graph space.

Lemma E *For all $n, m \in \mathbf{N}^+$ we have $\lambda(n) + \lambda(m) \leq \lambda(nm) \leq \lambda(n) + \lambda(m) + 1$. If n or m is a power of 2 then $\lambda(n) + \lambda(m) = \lambda(nm)$.*

Proof By definition we have $2^{\lambda(n)} \leq n < 2^{\lambda(n)+1}$ and $2^{\lambda(m)} \leq m < 2^{\lambda(m)+1}$ and hence $2^{\lambda(n)+\lambda(m)} \leq nm < 2^{\lambda(n)+\lambda(m)+2}$. Applying λ to each value yields the required result. Without loss of generality, if we assume that n is a power of two then $2^{\lambda(n)} = n$ and hence $2^{\lambda(n)+\lambda(m)} \leq nm < 2^{\lambda(n)+\lambda(m)+1}$. Once again applying λ to each value yields the required result. □

Theorem F *For all optimal formal addition chains for an arbitrary target n each with associated reduced graph $G = (V, E, \alpha, \omega)$ we have $\forall v \in V, d^-(v) \leq S(n) + 2, d^+(v) \leq S(n) + 2$.*

Proof We will prove the result for $\forall v \in V, d^+(v)$ and an examination of the dual graph will extend the proof to $d^-(v)$.

We consider an arbitrary optimal addition chain A for n and its associated reduced graph $G = (V, E, \alpha, \omega)$. Now select an arbitrary vertex $v \in V$ corresponding to chain element a_x. The path count of vertex v is the sum of $q = d^+(v) \geq 2$ path counts labeled in non-decreasing value as $w_1 \leq w_2 \leq \cdots \leq w_q$. Hence $a_x = \sum_{i=1}^{q} w_i$. From Theorems A and B in an optimal chain we may have at most 3 w_i that are equal, the rest being distinct. We may maximize our sum by setting $w_1 < w_2 < \cdots <$

$w_{q-2} = w_{q-1} = w_q$. Any other sum selecting a different element to duplicate (or not duplicating any element) must be less than this one. For each w_i value there is a corresponding addition chain value $a_{b_i} = w_i (1 \leq i \leq q)$. The strict ordering of the w_i values requires that $b_1 < b_2 < \cdots < b_{q-2} = b_{q-1} = b_q$. Since elements of an addition chain can at most be doubled at each step, we know that $a_{b_i} \leq 2^{b_i} (1 \leq i \leq q)$. The sum $a_x = \sum_{i=1}^{q} w_i \leq \sum_{i=1}^{q} 2^{b_i}$ is maximized by selecting adjacent chain elements butted next to a_{b_q} so that $b_i = b_q + i - q + 2 (1 \leq i \leq q-2)$ giving $a_x \leq \sum_{i=1}^{q} 2^{b_i} \leq 2^{b_q+1} + \sum_{i=1}^{q-2} 2^{b_q+i-q+2} = 2^{b_q-q+2}(2^q - 2)$. Since $S(n) \geq S_A(a_x) = x - \lambda(a_x)$ and $x \geq b_q + q - 1$ hence $S(n) \geq b_q + q - 1 - \lambda(2^{b_q-q+2}(2^q-2)) = q - 2$. □

Corollary *To bound the in-degree of a vertex v in a reduced graph in* [16,14] *the authors used the bound $a_x \leq wq$ where $w = \max\{p(\alpha(e)) : e \in E, \omega(e) = v\}$ and a_x is the addition chain element corresponding to vertex v. This can be used to place a lower bound on the number of small steps the construction of a particular vertex must add to an addition chain. $S_A(a_x, a_{b_q}) = x - b_q - (\lambda(a_x) - \lambda(a_{b_q})) \geq q - 1 - (\lambda(wq) - \lambda(w))$. From Lemma E we have $\lambda(wq) \leq \lambda(w) + \lambda(q)$ and if q is a power of 2 then $\lambda(wq) = \lambda(w) + \lambda(q)$. These may be combined to form $\lambda(wq) \leq \lambda(w) + \lceil \log_2(q) \rceil$. This yields $S_A(a_x, a_{b_q}) \geq q - \lceil \log_2(q) \rceil - 1$. Clearly then, as q grows the number of small steps grows and the chain becomes inefficient. A possible heuristic is to search for graphs with small q first and this is borne out in experiments.*

8 The effect of deleting addition chain elements

We now investigate the effect of deleting an addition chain element. We will find that deleting elements can only make the small step counts of chains stay the same or reduce. This observation helps answer questions of 'best possible' target values for addition chains with certain requirements.

Lemma G *If a formal addition chain A with elements $1 = a_0 \leq \cdots \leq a_k \leq \cdots \leq a_j \leq \cdots \leq a_i = a_j + a_k \leq \cdots \leq a_r = n$, $r \geq i > j \geq k \geq 0$ is converted to an addition chain A' for target n' by changing all elements of the chain that use a_i to instead use a_j and deleting the element a_i, then $2n' \geq n$ and hence $S_A(n) \geq S_{A'}(n')$. More formally if the elements of A' are $a'_l (0 \leq l < r)$ then:*

$$a'_l = \begin{cases} a_l, & \text{if } l < i \\ a'_j + a'_j & \text{if } i \leq l \leq r-1, \gamma(l+1) = i \text{ and } \delta(l+1) = i \\ a'_{\gamma(l+1)} + a'_j, & \text{if } i \leq l \leq r-1, \delta(l+1) = i \\ a'_j + a'_{\delta(l+1)}, & \text{if } i \leq l \leq r-1, \gamma(l+1) = i \\ a'_{\gamma(l+1)} + a'_{\delta(l+1)}, & \text{otherwise} \end{cases}$$

It should be pointed out that the functions γ and δ relate to the addition chain A and are used in the formation of addition chain A'.

We give an example of an optimal addition chain for 19 where we delete the element 3:

$$1, 2, 3, 6, 9, 18, 19 \Rightarrow 1, 2, 4, 6, 12, 13$$

In this example we obtain the value 4 from the value 6 by observing that $6 = 3 + 3$ and all usages of 3 have been replaced by 2. Similarly we obtain the value 6 from the value 9 by observing that $9 = 6 + 3$ and 6 has been replaced by 4 and 3 has been replaced by 2 and so on.

Proof We define a sequence of coefficients $C_m =< c_{m,l} : 0 \le l \le r > \ (0 \le m \le r)$ such that $a_m = \sum_{l=0}^{r} c_{m,l} a_l$. We will fix all $c_{m,l} = 0$ for $l > m$. We will show how to build the sequences such that they reflect the chain structure from a particular point and can then show how the chain values change when we delete a given element. For $0 \le m < i$ we set $c_{m,l} = \begin{cases} 0, & l \neq m \\ 1, & l = m \end{cases}$ for all $0 \le l \le r$. Clearly $c_{m,l} \ge 0 (0 \le m < i, 0 \le l \le r)$. For $i \le m \le r$ we set $c_{m,l} = c_{\gamma(m),l} + c_{\delta(m),l} \ge 0$ for $0 \le l \le r$ and hence $a_m = \sum_{l=0}^{r} c_{m,l} a_l$. Hence we define each addition chain element $a_m (j < m \le r)$ as a non-negative linear combination of the elements $a_p (0 \le p \le r)$ in such a way as to reflect chain construction. Hence we may express n as a linear combination of the terms $n = \sum_{l=0}^{r} c_{r,l} a_l = T + c_{r,j} a_j + c_{r,i} a_i$ where $T = \sum_{l=0}^{r, l \neq i, l \neq j} c_{r,l} a_l$ is the sum of all the other terms. When we replace all the usages of a_i by a_j we know exactly how this will change the target value n' if we know the linear combination coefficients $c_{r,j}$ and $c_{r,i}$. We obtain $n' = T + c_{r,j} a_j + c_{r,i} a_j$ giving $2n' - n = T + c_{r,j} a_j + c_{r,i}(2a_j - a_i)$. Since $a_i \le 2a_j$ we have $2n' - n \ge 0$. □

The coefficient construction used in Lemma G is made clear by viewing an addition chain symbolically and expanding terms that are above the truncation point. For example consider the addition chain

$$a_0 = 1, a_1 = 2, a_2 = 4, a_3 = 8, a_4 = 16, a_5 = 20, a_6 = 36, a_7 = 37, a_8 = 57.$$

We can set the truncation point at $i = 4$ and then substitute

$$a_8 = a_7 + a_5, a_7 = a_6 + a_0, a_6 = a_5 + a_4, a_5 = a_4 + a_2.$$

This yields:

$$a_0 = 1, a_1 = 2, a_2 = 4, a_3 = 8, a_4 = 16, a_5 = 20 = a_2 + a_4, a_6 = 36 = a_2 + 2a_4,$$
$$a_7 = 37 = a_0 + a_2 + 2a_4, a_8 = 57 = a_0 + 2a_2 + 3a_4.$$

9 Graph rearrangements

A rearrangement from one graph form to another is now shown. The patterns transformed are quite common in optimal addition chains and hence this rearrangement

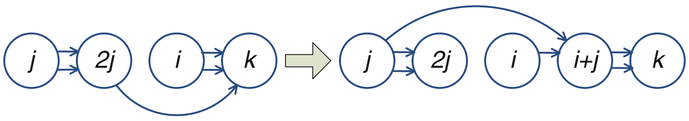

Fig. 6 Graph rearrangement used in Theorem H

has a significant impact on search times. See Fig. 6 for a graphical view of the transformation.

Theorem H *In the set of formal optimal addition chains for an arbitrary n of length r there must exist at least one addition chain A with associated reduced graph* $G = (V, E, \alpha, \omega)$ *with no three distinct arcs terminating at the same vertex* $(e, f, g \in E, \omega(e) = \omega(f) = \omega(g))$ *with at least two parallel* $(\alpha(e) = \alpha(f))$ *such that* $d^+(\alpha(g)) = 2$ *and* $\exists h, l \in E, \alpha(h) = \alpha(l), \omega(h) = \omega(l) = \alpha(g)$.

Proof We select an arbitrary optimal formal addition chain for n and look at its graph for the described pattern. If it does not have the pattern then the assertion that at least one addition chain does not have it is true. Otherwise it must have a match for this pattern in one or more places and we can pick the pattern whose e, f arcs attach to the largest vertex $\omega(e)$ in order of path count. We may have the pattern match more than once for the same vertex in which case we pick one at random. We can then eliminate this pattern in this one place by considering the following addition chain expanded from the graph $1, \ldots, a_j, \ldots, 2a_j, \ldots, a_i, 2a_i, 2a_i + 2a_j, \ldots, n$ with $a_i = p(\alpha(f)), 2a_j = p(\alpha(g))$. We can form the alternate chain $1, \ldots, a_j, \ldots, 2a_j, \ldots, a_i, a_i + a_j, 2a_i + 2a_j, \ldots, n$ which does not have the pattern beyond any vertex with a path count $> p(\omega(e))$. The number of matches to the pattern for the vertex $\omega(e)$ has been reduced by 1. It can be seen that the chain lengths remain the same by considering that both the arc and vertex count increase by 1 keeping $|E| - |V| + 1$ invariant. We can repeat this operation until the graph does not contain the pattern. □

10 Enumerating reduced graphs

We can deconstruct a reduced graph $G = (V, E, \alpha, \omega)$ in path count order $(v_0, \ldots, v_{|V|-1} \in V)$ with $|V| > 1$ associated with an optimal formal addition chain by producing a graph minor by deleting the last vertex $v_{|V|-1}$ and its associated arcs. By repeating this process we eventually reach the graph containing only the single source vertex v_0. The intermediate graphs will not necessarily be reduced graphs because they may not satisfy the out-degree requirements as we have deleted arcs. We will call such a graph *deficient* and talk of the set of deficient vertices as the set of vertices whose out-degree is too small. We can reverse this process starting with the vertex v_0 representing 1 and creating all possible vertices v_1 by adding 2 or more arcs to the existing graph. Recursively doing this can enable us to enumerate all possible reduced graphs with prescribed properties (for numbers less than particular bounds or with a particular number of small steps). We can limit the search space extensively by not enumerating any graphs with known patterns described above that would make them non-optimal. We can also avoid any graphs that could be rearranged to other forms as we will search their space in different branches of the tree. As mentioned earlier, we will generate deficient graphs in our search. By tracking all the deficient vertices and incorporating them into a simple bound to be described next, we can prune well in advance of their usage.

In [11] an extensive set of bounds was developed to make searching for addition chains efficient. We will take the most trivial bound (class A) but use it in a way that is reminiscent of the slant bounds developed in the same paper. We will focus on enumerating all addition chain targets n within a particular range $L \le n \le U$ with $S(n) \le s$. Other possibilities could be fruitful but have not been investigated.

A *bounding sequence* $B(U, s) =< b_0, b_1, \ldots, b_{\lambda(U)+s} >$ is constructed so that the partial addition chain $a_0, a_1, \ldots, a_i$ only has to be considered further if $a_i \ge b_i$. Since we can convert our partial reduced graphs to addition chains as we build them if we have a suitable bounding sequence, we can use this to prune graphs from our search. We may initialize our bounding sequence with the following simple Theorem:

Theorem I *When enumerating all optimal addition chains for $n \le U$ with $S(n) \le s$ a suitable first bounding sequence is $B(U, s) =< b_0, b_1, \ldots, b_{\lambda(U)+s} >$ defined by $b_i = 2^{i-s} (0 \le i \le \lambda(U) + s)$.*

Proof For an arbitrary addition chain A, if at position i we have $a_i < b_i = 2^{i-s}$ then we must have $i \ge s$ since $0 < a_i$. From this we obtain $S_A(a_i) = i - \lambda(a_i) > i - \lambda(b_i) = s$. So any resulting chain would have too many small steps to be valid for the enumeration. Since we are only enumerating addition chains for targets $n \le U$ then if $n = a_i$ occurred in a chain at position $i > \lambda(U) + s$ then $S_A(a_i) = i - \lambda(a_i) > \lambda(U) + s - \lambda(U) = s$ and hence this chain would have too many small steps to be part of the enumeration. $\square$

We may improve the bounds (make them tighter) and hence improve the runtime of the enumeration by changing them in response to targets of addition chains found. We define a function that keeps track of addition chains found so far

$$\rho(n, s) = \begin{cases} 1, & S(n) \le s \text{ is known} \\ 0, & S(n) \text{ is unknown} \end{cases} .$$

Clearly this could be implemented as a simple bitmap. We can now show a simple method of updating the bounds at runtime as new targets are discovered:

Theorem J *When enumerating all optimal addition chains for $n \le U$ with $S(n) \le s$ a bounding sequence $B(U, s) =< b_0, b_1, \ldots, b_{\lambda(U)+s} >$ may be improved for an arbitrary element $b_i \leftarrow b'_i = b_i + 1$ if $\rho(b_i, s) = 1$ and $i = \lambda(U) + s$ or $2b_i < b_{i+1}$ or $b_{i+1} > U$.*

Proof The new bounds affect only partial addition chains that begin $a_0, \ldots, a_i = b_i, \ldots, a_r = n$. These would be accepted by the old bound and rejected by the new bound. There are two cases to consider. The first case we may have is $i = r$ and hence $b_i = n$. This case is the only case to consider if $i = \lambda(U) + s$ or $b_{i+1} > U$ as this partial chain must be complete. Since $\rho(b_i, s) = 1$ we have already noted this target in our enumeration and we do not need to examine it again. The second case is $i < r$ but since $a_{i+1} \leq 2a_i = 2b_i < b_{i+1}$ we know that any continuation of this addition chain will be pruned. □

A bounding sequence as defined by Theorems I and J can be used to prune a partial chain with a 'best possible' set of targets. For example, if it is known that some constraints in the construction of a chain limit the upper bounds of addition chain values at different positions and that all chains must be long enough to reach the positions, then these 'best possible' targets can be pruned with the previously described bounding sequences. This is shown in the following Theorem.

Theorem K *A is an arbitrary optimal addition chain of length r for target n truncated at position* $0 < t < r$. *We have a bounding sequence* $B(U, s) =< b_0, b_1, \ldots, b_{\lambda(U)+s} >$ *as defined in Theorems I and J. If we know that from some position p with* $t < p \leq r$ *that* $a_r \leq 2^{r-p} m_p$ *and* $m_p < b_p$ *then this partial chain may be pruned away.*

Proof Since we have $m_p < b_p$ and from Theorems I and J we have $2b_p \leq b_{p+1}$ or $b_{p+1} > U$ we know that $a_r \leq 2^{r-p} m_p < 2^{r-p} b_p \leq b_r$ or $a_r > U$. In either case this partial addition chain cannot lead to anything new. □

11 Pruning partial addition chains based on subsequent element usage

As mentioned above, during graph enumeration we will likely generate deficient graphs. These graphs can be converted to partial addition chains that require future elements to use particular values some minimal number of times to be valid. These additional references satisfy the minimum degree requirements of the reduced graph. We will now show how to incorporate the information about the deficient graph into two variables and use this to prune the partial chains and hence the partial graphs at a much earlier point in the enumeration.

Lemma L *An optimal addition chain A of length r for an arbitrary target n is truncated at position t* $(0 \leq t < r)$. *The partial chain is thus* $a_0, \ldots, a_t$. *If we know that future elements of the addition chain must use certain elements* (a_{v_i}, $1 \leq i \leq f$, $f \geq 2$ *not necessarily distinct) from the partial chain in their sum, then* $n \leq 2^{r-t-f+1} \sum_{i=1}^{i \leq f} a_{v_i}$ *and* $S(n) \geq t + f - 1 - \lambda(\sum_{i=1}^{i \leq f} a_{v_i})$. *In essence, our target number is maximized by using all the required elements immediately rather than delaying their use and can be represented by the 'best possible' chain* $a_0, \ldots a_t, a_{v_1} + a_{v_2}, a_{v_1} + a_{v_2} + a_{v_3}, \ldots, \sum_{i=1}^{i \leq f} a_{v_i}$ *followed by doublings.*

Proof We will show this result by taking a counter example chain A truncated at position $t (0 \leq t < r)$ with smallest length r and largest target n amongst those of the same length. The chain after the truncation point requires the use of not necessarily distinct elements a_{v_i}, $1 \leq i \leq f$, $f \geq 2$ called *back references*. We say that a back reference is satisfied if it is used in the construction of an element. A counter example must have $n > 2^{r-t-f+1} \sum_{i=1}^{i \leq f} a_{v_i}$. Converting to small step counts on each side will yield $S(n) < t + f - 1 - \lambda(\sum_{i=1}^{i \le f} a_{v_i})$. By examining each possible way a_{t+1} may be constructed we will show that an alternate chain A' with target n' conforming to the construction of the 'best possible' chain has either $n' \ge n$ with length r or $2n' \ge n$ with length $r - 1$ in contradiction to the counter example selection. If $t + 1 < r$ we may move the truncation point to position $t + 1$ by looking at the construction of element a_{t+1}. We form a new list of back references by removing those satisfied by the construction of a_{t+1}. If all back references have been satisfied then the Lemma is trivially true since $f = 2$ which requires $a_{t+1} = a_{v_1} + a_{v_2}$and the rest of the chain can do no better than doubling the last element repeatedly. If all back references have not been satisfied we must add a new back reference for a_{t+1} since an optimal chain must use all elements except the last. We may form element a_{t+1} in one of 4 ways:

$$a_{t+1} = \begin{cases} a_{v_j} + a_{v_k}(1 \le j, k \le f, j \ne k) & \text{Type 1} \\ a_j + a_k(a_j, a_k \notin \{a_{v_i} : 1 \le i \le f\}) & \text{Type 2} \\ a_{v_j} + a_t(1 \le j \le f) & \text{Type 3} \\ a_{v_j} + a_k(1 \le j \le f, 1 \le k < t) & \text{Type 4} \end{cases}$$

Type 1:

Element a_{t+1}'s construction follows the best possible pattern described above and we may examine the same chain truncated at element $t + 1$ instead. A valid counter example must therefore have a step of type > 1.

Type 2:

Element a_{t+1}'s construction satisfies no back references and so it may be deleted to form a new addition chain A' with target n'. By Lemma G the new target n' for the chain of length $r - 1$ must have $2n' \ge n$. This is in contradiction to the counter example selection. A valid counter example must therefore have a step of type > 2.

Type 3:

An optimal chain must make use of all elements except the last. Therefore we must have $a_t = a_{v_k}$ for at least one $1 \le k \le f$. In order not to be a step of type 1 we would require $j = k$ and hence there is only one back reference to a_{v_j} satisfied by step $t + 1$. An optimal chain must use element a_{t+1} in some future step $a_s = a_{t+1} + a_u$ with $t + 1 < s \le r, 0 \le u \le r$. We may form a new addition chain A' with target n' and elements $a'_i(0 \le i \le r - 1)$ by deleting element a_{t+1} causing $a'_{s-1} = a'_t + a'_v$ which will satisfy the back reference satisfied by step $t + 1$ later. By Lemma G the new target n' for the chain of length $r - 1$ must have $2n' \ge n$. This is in contradiction to the counter example selection. A valid counter example must therefore have a step of type > 3.

Type 4:

We have $a_{t+1} = a_{v_j} + a_k(1 \le j \le f, 1 \le k < t)$. We may form a new addition chain A' with target n' with length r and elements $a'_i(0 \le i \le r)$ by instead forming $a'_{t+1} = a'_{v_j} + a'_t$. This new chain must have $n' \ge n$ since $a_k < a_t$ which is a

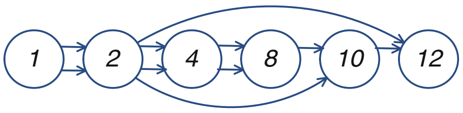

Fig. 7 A graph with deficient vertices 1, 8, 10 and 12

contradiction to the counter example selection. This has exhausted all chain element types and the Lemma is proved. □

Example Let us assume that in the partial addition chain 1, 2, 4, 8, 10, 12 obtained while searching for numbers requiring 3 small steps we somehow know that chain elements 1,8,10 and 12 all need additional references. Lemma L tells us that the 'best possible' optimal chain under these constraints is 1, 2, 4, 8, 10, 12, 22, 30, 31, ..., $2^d 31$ which has 4 small steps. This partial chain can be rejected at this point even though there are a number of ways to continue the chain within the 3 small step limit. The eventual use of the required chain elements will always create too many small steps.

Remark This method along with Lemma G and similar arguments to Theorem F can be used to show that if the computation of an optimal addition chain for n requires t temporary variables then $S(n) \geq t - 1$.

Theorem M *An optimal addition chain A for target $n > 1$ with reduced graph $G = (V, E, \alpha, \omega)$ can be truncated by removing all vertices with path counts greater than some threshold $n > T > 1$ and their associated arcs to create the graph G' with an associated addition chain A'. This creates a deficient reduced graph. If we label the not necessarily unique deficient vertices $v_1, \ldots, v_f$, $f \geq 2$ then we have $n \leq 2^{r-r'-f+1} \sum_{i=1}^{i \leq f} p(v_i)$ where $r = |E| - |V| + 1$ and $r' = |E'| - |V'| + 1$ are the lengths of the associated addition chains of A and A' respectively.*

Proof Some vertices have been removed because $n > T$ so at a minimum the last vertex (by path count) has been removed. The vertex in G' with largest path count will need at least two arcs to satisfy the requirements of a reduced graph. This means that G' is deficient and we can label the deficient vertices $v_1, \ldots, v_f$, $f \geq 2$. From Lemma L we have $n \leq 2^{r-r'-f+1} \sum_{i=1}^{i \leq f} p(v_i)$. □

We return to the example given for Lemma L and show how this example is derived from a deficient graph that is seen while enumerating 3 small step numbers. Fig. 7 shows a deficient reduced graph. The deficient vertices are 8 and 10 because they only have out-degrees of 1. This will cause additional vertices of greater path counts to be required making vertex 12 deficient. Theorem C tells us we need an out-degree of at least 3 for vertex 1 making that deficient also.

12 Algorithm

A backtracking algorithm will now be described to calculate all possible numbers within an interval ($[L, U]$) that have addition chains with s small steps. The algorithm will operate on the following variables:

List of ordered vertices $V =< v_0, \ldots, v_w >$ in path count order. Bounding sequence $B(U, s) =< b_0, b_1, \ldots, b_{\lambda(U)+s} >$ maintained as per Theorems I and J.

List of addition chain elements $A =< a_0, \ldots, a_r >$ not necessarily ordered. c the count including multiplicities of the number of deficient vertices in V. t the sum of the path counts of the deficient vertices in V.

We initialize the variables and begin enumerating as follows:

```
procedure EnumerateReachableNumbers (s):
begin
  $t \leftarrow 0$;  $c \leftarrow 0$;
  if $s > 0$ then $t \leftarrow 1$;  $c \leftarrow 1$; » we need three references to 1
  $w \leftarrow 0$;  $r \leftarrow 0$;
  $V \leftarrow< v_0 >$;  $a_0 \leftarrow 1$;  $A \leftarrow< a_0 >$,  $p(v_0) \leftarrow 1$;
  RecordAndAddNewVertex(s, V, A, t, c, w, r);
end
```

This initialization corresponds to the simplest optimal addition chain for the target 1.

The main driving force in the program takes the existing list of vertices, records the target reached if needed, and transitions it to a new list with an additional vertex v_{w+1}.

```
procedure RecordAndAddNewVertex ($s, V, A, t, c, w, r$) :
begin
  » if not deficient, record we reached $a_r$ and update the bounding sequence if
  needed
  if $c = 0$ and $a_r \geq L$ and $a_r \leq U$ then
  begin
    $\rho(a_r) \leftarrow 1$;
    Update bounding sequence $B$ if needed;
  end
  » record the fact that the vertex $v_w$ requires at least two usages to not be deficient

                    $c \leftarrow c + 2$;  $t \leftarrow t + 2p(v_w)$;

  » we need four references to 1 for the special case of graphs starting 1,3
  if $s > 1$ and $w = 1$ and $p(v_w) = 3$ then $c \leftarrow c + 1$;  $t \leftarrow t + 1$;
  If $r + c - 1 > \lambda(U) + s$ then return as this chain would be too long
  If $t < b_{r+c-1}$ and $t \leq U$ then return as this chain cannot reach anything new
  » loop over all possible in-degree values of a new vertex
  for $q \leftarrow 2$ until $\min(s + 2, \lambda(U) + s - r + 1$ do
  begin
    Select $q$ not necessarily unique vertices avoiding the patterns described earlier;
    » Avoid selecting the same vertex more than three times.
    » Avoid selecting any two vertices multiple times.
    » Avoid parallel arcs from more than one vertex.
    » Avoid parallel arcs to a vertex that already has parallel arcs to another vertex.
    » Avoid creating the pattern described in Theorem H.
```

```
            » Avoid creating a new vertex that is smaller than the previous or out of the
            bounds $[L, U]$.
            Form a new vertex $v_{w+1}$ by adding arcs to each selected vertex;
            Form $a_{r+1}, \ldots, a_{r+q-1}$ by taking the sums of the path counts of the selected
            vertices in an arbitrary order;

            $A' \leftarrow A+ < a_{r+1}, \ldots, a_{r+q-1} >$;
            $V' \leftarrow V+ < v_{w+1} >$;

            Form $t', c'$ from $t, c$ by subtracting out any vertex that was deficient and was
            used by the vertex selection;
            RecordAndAddNewVertex $(s, V', A', t', c', w+1, r+q-1)$;
        end
    end
```

13 Results

The enumeration algorithm was used to find all $l(n)$ with $n \leq 2^{32}$ by enumerating all n with $S(n) \leq 7$ and doing limited enumeration to fill in the gaps with $S(n) = 8$. The runtime for this calculation was in the order of a month using 12 (two quad and two dual) 2.66 GHz processors. Previously, using a number of techniques outlined in this paper to speed up the fastest known algorithm [16,14], a run time of 1.5 years was needed on similar hardware to find all $l(n)$ with $n \leq 2^{26}$. Clearly there must be a huge amount of redundancy in calculating optimal addition chains for a single value currently based on these numbers.

By modifying the bounding sequence defined in Theorems I and J such that $\rho(n, s) = \begin{cases} 1, \text{ if } v(n) \leq 2^s \\ 0, \text{ otherwise} \end{cases}$ it is possible to quickly search for counter examples to the Knuth–Stolarsky conjecture [4, p. 470, 17]. This conjecture essentially asserts that $S(n) \geq \log_2 v(n)$ but the inequality appears in two different formats. No counter examples were found for $n \leq 2^{64}$.

In [18] Goulard asked if $l(2n) = l(n) + 1$ and this was answered in the positive in [19] but only verbal arguments were made. The conjecture appears again in [20] before a counter example ($l(191) = l(382)$) was found with the aid of a computer by Knuth [21]. Thurber proved there exist infinite sequences of n with $l(2n) = l(n)$ in [22,23]. In [24] Granville asks if there is an n with $l(n) = l(2n) = l(4n)$ and an equivalent question appeared as a gap in position 4 in the sequence A115016 of [25]. We found the first example for this with $n = 30958077$. At higher values, the first example where $l(2n) < l(n)$ was found with $n = 375494703$, $34 = l(2n) < l(n) = 35$.

The Scholz–Brauer conjecture [3,26] asserts that $l(2^n - 1) \leq l(n) + n - 1$. Applying the Theorems described above to the algorithms in [11,16,14] we were able to show the conjecture was true for $n < 5784689$. This was achieved by showing all numbers below this bound were so called Hansen numbers [4, pp 479, 10]. Strict equality was noted in [17] as the only case seen for $n \leq 8$ and this was extended to $n \leq 24$ and $n = 32$ in [27]. Computer calculations confirmed equality for $n \leq 28$ [11,14,28] and we confirmed equality for $n \leq 64$.

Table 2 New $c(r)$ and $d(r)$ values

r	$c(r)$	r	$c(r)$	r	$d(r)$	r	$d(r)$
29	7624319	36	550040063	26	1896704	33	140588339
30	14143037	37	994660991	27	3501029	34	261070184
31	25450463	38	1886023151	28	6465774	35	485074788
32	46444543	39	3502562143	29	11947258	36	901751654
33	89209343			30	22087489	37	1677060520
34	155691199			31	40886910	38	3119775195
35	298695487			32	75763102		

New techniques that allow the boundaries of what can be calculated to be enlarged allow insight into how certain functions grow. Table 2 shows new values for $d(r) = |\{n : l(n) = r\}|$ (the total number of integers with a particular sized optimal addition chain) and $c(r) = \min(\{n : l(n) = r\})$ (the smallest number with a particular sized optimal addition chain).

These new values are in accordance with the conjecture that $c(r) \sim 2^{r-\frac{r}{\log_2 r}}$ given in [14].

14 Conclusions and future research

Calculating optimal addition chains for multiple different numbers appears to contain a great deal of overlapping calculations as we gain considerable speed ups by calculating them simultaneously. Exploiting required chain structures to prune early also has a very large effect on runtimes. The location of the first non-Hansen number suggests that attempts to prove the Scholz–Brauer conjecture by proving that among the optimal addition chains there exists at least one Hansen chain cannot be successful in the long run [29]. The Scholz–Brauer conjecture needs to be proved for non-Hansen numbers to bridge the exposed gap. The non-Hansen number and the case $l(2n) < l(n)$ appear to be part of infinite sequences of such numbers and attempts to prove this might be fruitful. Another interesting question might be to ask if $l(n)$ and $l(2n)$ can become arbitrarily far apart. There may be much scope to find more graph rearrangements as well as allowed and denied graph structures to further improve runtimes. Can the bounding sequences of Sect. 10 be improved in general or by looking at the remaining target numbers to be found? Can the linear bound of the in-degree in Sect. 7 be improved? The work of Hansen proving that the binary method is optimal [10] for numbers with large runs of zeros in their binary representations suggests we can only improve the bounds based on the structure of the target.

Acknowledgments This paper benefited greatly from the suggestions of Andrew Rogers, Christine Woskett, Dan Eilers, Ed Thurber, Ilya Mironov and Tomasz Ostwald.

References

1. Dellac H (1894) Question 49. L'Intermédiaire Math 1:20
2. de Jonquiéres E (1894) Question 49 (H. Dellac). L'Intermédiaire Math 1(20):162–164
3. Scholz A (1937) Aufgabe 253. Jahresbericht der deutschen Mathematiker-Vereingung 47(2):41–42
4. Knuth DE (1997) The art of computer programming V. 2 Seminumerical algorithms, 3rd edn. Addison-Wesly, Reading, pp 461–485 (ISBN 0-201-89684-2)
5. Gioia AA, Subbarao MV, Sugunamma M (1978) The Scholz–Brauer Problem in Addition Chains II. In: Proc. Eighth Manitoba Conference on Numerical Math. and Computing, XXII, pp 174–251
6. ElGamal T (1985) A public-key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans Inform Theory IT-31(4):469–472
7. FIPS Publication 186, February 1, 1993, Digital Signature Standard
8. Gordon DM (1998) A survey of fast exponentiation methods. J Algorithms 122:129–146
9. Osorio-Hernández L, Mezura-Montes E, Cruz-Cortés N, Rodríguez-Henríquez F (2009) A genetic algorithm with repair and local search mechanisms able to find minimal length addition chains for small exponents. In: IEEE congress on evolutionary computation (CEC) 2009, Trondheim, Norway May 18–21, 2009, IEEE Computer Society, 8 p
10. Hansen W (1957) Untersuchungen über die Scholz–Brauerschen Additionsketten und deren Verallgemeinerung. Göttingen, Wiss. Prüfungsamt, Schriftl. Hausarbeit zum Staatsexamen
11. Thurber EG (1999) Efficient generation of minimal length addition chains. SIAM J Comput 28:1247–1263
12. Pippenger N (1976) On the evaluation of powers and related problems. In: Proceedings, 19th Annual IEEE Symposium on the foundations of computer science, Houston TX, pp 258–263
13. Lupanov OB (1956) Diode and contact-diode circuits. Dokl AN SSSR 111(6):1171–1174
14. Bleichenbacher D, Flammenkamp A (1997) An efficient algorithm for computing shortest addition chains. http://www.uni-bielefeld.de/~achim/ac.dvi
15. Flammenkamp A (1999) Integers with a small number of minimal addition chains. Discrete Math 205:221–227
16. Bleichenbacher D (1996) Efficiency and Security of cryptosystems based on number theory. Dissertation to Swiss Federal Institute of Technology Zürich, ETH Zürich
17. Stolarsky KB (1969) A lower bound for the Scholz–Brauer problem. Can J Math 21:675–683
18. Goulard A (1894) Question 393. L'Intermédiaire Math 1:234
19. de Jonquiéres E (1895) Question 393 (A. Goulard). L'Intermédiaire Math 2:125–126
20. Utz WR (1953) A note on the Scholz–Brauer problem in addition chains. In: Proceedings of the American Mathematical Society, vol 4, pp 462–463
21. Knuth DE (1964) Addition chains and the evaluation of n-th powers. Notices Am Math Soc 11:230–231
22. Thurber EG (1971) The Scholz–Brauer Problem for addition chains. Ph.D. Thesis, University of Southern California
23. Thurber EG (1973) The Scholz–Brauer problem on addition-chains. Pacific J Math 49:229–242
24. Guy RK (1994) Unsolved problems in number theory, 2nd edn. Springer, New York
25. The On-Line Encyclopedia of Integer Sequences, http://www.research.att.com/~njas/sequences
26. Brauer AT (1939) On addition chains. Bull Am Math Soc 45:736–739
27. Thurber EG (1976) Addition chains and solutions of $l(2n) = l(n)$ and $l(2^{n-1}) = n + l(n) - 1$. Discrete Math 16:279–289
28. Thurber EG Private communication
29. Bahig HM, Nakamula K (2002) Some properties of nonstar steps in addition chains and new cases where the Scholz conjecture is true. J Algorithms 42(2):304–316