Nonnegative Integral Subset Representations of Integer Sets

Michael J. Collins * David Kempe † Jared Saia ‡ Maxwell Young‡

Abstract

We consider an integer-subset representation problem motivated by a medical application in radiation therapy. We prove NP-completeness, derive nontrivial bounds, and report on the performance of a greedy heuristic.

Keywords: approximation algorithms; combinatorial problems; heuristics; additive number theory; radiology

1 Introduction

Given a set of positive integers $S = \{s_1 < s_2 < \cdots < s_n\}$, the GENERATING SET problem consists of finding a minimum size set of positive integers $T = \{t_1 < t_2 < \cdots < t_m\}$ such that every element of S is the sum of a subset of T. Such a set T is called a "generating set" for S; note that T does *not* have to be a subset of S. In addition to being a very natural problem in combinatorial number theory, this problem is motivated by medical applications in planning radiation therapy: elements of S represent radiation dosages required at various points, while an element of T represents a dose delivered simultaneously to multiple points. We seek to minimize the number of steps required to deliver exactly the specified dose to each point. There is no need to allow T to be a multiset; if some t appears twice in T, we can w.l.o.g. replace the second appearance with $2t$. Somewhat similar optimization problems arising from radiation therapy are considered in [1, 3, 4]. The case in which the elements of T can be negative or fractional is considered in [2, 5, 6]. Trivially $\lceil \log n \rceil \leq m \leq \min(n, \lceil \log s_n \rceil)$, since T must be large enough to have n distinct subsets, and S can be represented by itself or by $\{1, 2, 4, \ldots 2^{\lfloor \log s_n \rfloor}\}$.

*Sandia National Laboratories, Albuquerque, NM, USA; email: mjcolli@sandia.gov

†Dept. of Computer Science, University of Southern California, CA, USA; email: dkempe@usc.edu

‡Dept. of Computer Science, University of New Mexico, NM, USA; email: saia@cs.unm.edu, young@cs.unm.edu

2 NP-Completeness

In this section, we prove the following decision version of the GENERATING SET problem to be NP-complete.

Definition 1. GENERATING SET*:*
Input: A target set S of positive integers, and a non-negative integer k.
Question: Is there a set T of positive integers such that $|T| = k$, and every element $s \in S$ is the sum of some subset of the elements in T?

We prove that GENERATING SET is NP-complete via a reduction from POSITIVE NAE-3SAT(5), a variant of NAE-3SAT, which is proven NP-complete in [7]. Reduction from SUBSET SUM might appear to be more natural, but no such reduction is known.

Definition 2. POSITIVE NAE-3SAT(5)*:*
Input: A SAT formula ϕ with $n \geq 5$ variables and m clauses. Each clause contains exactly 3 distinct and uncomplemented variables.
Question: Is there a truth assignment such that each clause of ϕ contains both a true variable and a false variable?

Theorem 1. GENERATING SET *is NP-complete.*

Our reduction takes an instance ϕ of POSITIVE NAE-3SAT(5), with variables labeled $x_1, \ldots, x_n$ and m clauses, and constructs an instance of GENERATING SET in the following way:

- Add 1 to S.
- For each variable x_i, add $a_i := n^i$ to S. We call a_i the "variable number" for x_i.
- For each clause c containing the variables x_i, x_j, x_k, add the $y_c := a_i + a_j + a_k - 1 = n^i + n^j + n^k - 1$ to S. We call y_c the "clause number" for c.

The decision question of GENERATING SET is then whether the set S can be represented by an integer set of size at most $n + 1$.

We begin by establishing useful lemmas about the possible structures of generating sets T. In the sequel, we assume that ϕ is fixed (and has $n \geq 5$ variables), and we always use S to refer to the set generated by the reduction. We also always assume that $T = \{1 \leq b_1 \leq b_2 \leq \ldots \leq b_n\}$ is a generating set for S of size $n + 1$.

Lemma 1. *For all $i \leq n$, we have $a_i - \sum_{j=1}^{i-1} a_j - 1 \leq b_i \leq a_i$.*

Proof. First notice that 1 must be in T. Second, assume that $b_i > a_i$ for some i, and let r be the smallest such index. Because $b_i \leq a_i$ for $i \leq r - 1$, we have

$1 + \sum_{i=1}^{r-1} b_i \leq 1 + \sum_{i=1}^{r-1} a_i < n^r = a_r$ (for $n \geq 2$). This implies that T cannot generate a_r, which is a contradiction. Thus, $b_i \leq a_i$ for all i.

Next, assume that $b_i < a_i - \sum_{j=1}^{i-1} a_j - 1$ for some i, and let r be the largest index such that $b_r < a_r - \sum_{i=1}^{r-1} a_i - 1$. Then, $b_{r+1} \geq a_{r+1} - \sum_{i=1}^{r} a_i - 1 > a_r$ (for $n \geq 3$). This implies that a_r is the sum of elements of some subset of $\{1, b_1, \ldots, b_r\}$. But $1 + \sum_{i=1}^{r} b_i \leq a_r - 1$, which means that T cannot generate a_r, which is a contradiction. Thus, for all i, $b_i \geq a_i - \sum_{j=1}^{i-1} a_j - 1$. ■

Lemma 2. *Let* $\mathbf{b} = (b_1, b_2, \ldots, b_n)$*, and* $X = \{0, 1, 2, 3\}^n$*. Then, for any two distinct vectors* $\mathbf{u}, \mathbf{v} \in X$*,* $|\mathbf{u} \cdot \mathbf{b} - \mathbf{v} \cdot \mathbf{b}| \geq 3$*.*

Proof. Let $\mathbf{u} = (u_1, u_2, \ldots, u_n)$ and $\mathbf{v} = (v_1, v_2, \ldots, v_n)$, and k be the largest index such that $u_k \neq v_k$. Without loss of generality, $u_k > v_k$. Then, using Lemma 1, $|\mathbf{u} \cdot \mathbf{b} - \mathbf{v} \cdot \mathbf{b}| \geq \mathbf{u} \cdot \mathbf{b} - \mathbf{v} \cdot \mathbf{b} \geq b_k - 3\sum_{i<k} b_i \geq (a_k - \sum_{i<k} a_i - 1) - 3\sum_{i<k} a_i = n^k - \frac{4n^k - 3n - 1}{n-1} \geq 3$, for $n \geq 5$. ■

Corollary 1. *For all* $i \leq n$*,* T *generates either* a_i *or* $a_i - 1$*, but not both, without using* 1*.*

Proof. Apply Lemma 2, taking $\mathbf{u}, \mathbf{v}$ to be the 0-1 vectors describing which elements of T are used to generate a_i and $a_i - 1$, and notice that $|\ a_i - (a_i - 1)| = 1$. ■

Lemma 3. *For any "clause number"* $y_c = a_i + a_j + a_k - 1$ *in* S*, the following two statements are true:*

1. *At least one of* $\{a_i, a_j, a_k\}$ *can be generated by* T *without using* 1*.*
2. *There is an* $a \in \{a_i, a_j, a_k\}$ *such that* T *generates* $a - 1$ *without using* 1*.*

Proof. If each of a_i, a_j, a_k requires 1 to be generated, then taking the corresponding incidence vectors $\mathbf{v_i}, \mathbf{v_j}, \mathbf{v_k}$ for the elements of T except 1, and writing $\mathbf{v} := \mathbf{v_i} + \mathbf{v_j} + \mathbf{v_k}$, we find that $\mathbf{v} \cdot \mathbf{b} = a_i + a_j + a_k - 3$. Now, by Lemma 2, T cannot generate $a_i + a_j + a_k - 2$ or $a_i + a_j + a_k - 1$ without the number 1, and hence cannot generate $a_i + a_j + a_k - 1$ at all, a contradiction. Similarly, if all of a_i, a_j, a_k can be generated without using the number 1, the analogous vector $\mathbf{u}$ satisfies $\mathbf{u} \cdot \mathbf{b} = a_i + a_j + a_k$. Again, by Lemma 2, $a_i + a_j + a_k - 1$ is not generated by T, a contradiction. ■

We are now ready to prove Theorem 1:

Proof of Theorem 1. It is clear that Generating Set is in NP, and the reduction clearly runs in polynomial time. To prove the correctness of the reduction, first assume that we have an assignment such that each clause of ϕ has either one or

two true variables. We create T in the following way. First, T always contains the number 1. In addition, for each i, if $x_i = \texttt{true}$ in the satisfying truth assignment, T contains the number n^i; otherwise, it contains the number $n^i - 1$. Clearly, T generates all "variable numbers" a_i, either simply as $a_i = n^i$, or as $a_i = (n^i - 1) + 1$. Since every clause c has either one or two true variables, summing the values corresponding to the variables will give either $a_i + a_j + a_k - 1$ or $a_i + a_j + a_k - 2$. In the former case, y_c is already generated, and in the latter case, it is generated by adding 1.

Conversely, if T is a generating set of size $n + 1$, we define a truth assignment to the x_i by setting $x_i = \texttt{true}$ if T generates a_i without using the number 1, and $x_i = \texttt{false}$ otherwise (in which case T generates a_i using the number 1, by Corollary 1). By Lemma 3, for each clause c, the fact that T can generate the value $y_c = a_i + a_j + a_k - 1$ guarantees that the truth assignment received by the corresponding clause contains at least one variable set to `true`, and at least one variable set to `false`, completing the proof. ■

3 Lower Bounds on m

The trivial lower bound from the introduction can be improved in some cases. Let $|T| = m$, and define the multiset of differences

$$D_S = \{s_j - s_i : i < j \leq n\}.$$

Let d_S be the number of distinct elements in D_S. At most $\frac{3^m-1}{2}$ positive differences can be generated as the difference between two subsets of T; in such a difference, each t_i appears positively, negatively, or not at all, and half of the nonzero differences are negative. Thus

$$m \geq \log_3 2d_S > 0.63 \log 2d_S$$

(where all logs are base 2 unless indicated otherwise). In particular, if $d_S = \alpha n^2/2$, we have

$$m > 1.26 \log n - 0.63 \log \frac{1}{\alpha}.$$

More generally, there are fewer than $(k+1)^m$ distinct sums of k subsets of T, so this must be larger than the number of distinct values of sums of k elements of S (allowing multisets). Then, we have

Theorem 2. *If there are $\alpha \frac{n^k}{k!}$ distinct sums of k elements of S and T generates S, then*

$$m > \frac{k}{\log(k+1)} \log n - \frac{\log k!}{\log(k+1)} - \frac{\log \alpha^{-1}}{\log(k+1)} \ .$$

We can also obtain lower bounds of $1 + \lceil \log n \rceil$ by considering parities. Let n' be the number of odd values in S; if $n' \neq 0$ then there must be at least one odd

element in T, so exactly 2^{m-1} subsets of T have odd sums. Thus, 2^{m-1} must be at least $\max(n', n-n')$, and

$$m \geq 1 + \lceil \log \max(n', n-n') \rceil \ .$$

More generally, consider the values in S mod M. If any s_i is not zero mod M, then some t_j is nonzero mod M, thus no more than half of the subsets of T can have the same value mod M (since adding or removing t_j changes the value of a set mod M), and $m \geq 1 + \lceil \log \hat{n} \rceil$ where $\hat{n}$ is the size of the largest equivalence class mod M.

4 Fraction of Instances Requiring Large Representations

If we are interested in instances whose numbers are bounded by 2^k, and n is sufficiently large relative to k, then most of these instances will not have small generating sets (i.e., sublinear in k). Specifically, there are at most $\binom{2^k}{j}$ sets T of size j, each generating at most 2^j distinct sums. The only n-element sets generated by T are its n-element subsets. Therefore, there are at most

$$\binom{2^k}{j} \cdot \binom{2^j}{n} \tag{1}$$

subsets of size n that can be generated by sets T of size at most j. On the other hand, the total number of problem instances of size n is $\binom{2^k}{n}$, and using the bounds $(\frac{a}{b})^b \leq \binom{a}{b} \leq (\frac{ea}{b})^b$, the fraction of instances that can be generated by sets of size at most j is at most

$$2^{jk+jn+(\log e)(j+n)-kn-j\log j}. \tag{2}$$

By taking limits for k, n, we obtain the following:

Theorem 3. *Fix $\alpha < 1$, and consider problem instances of n numbers bounded by 2^k.*

1. *If $\frac{n}{k} \to \infty$ as $k, n \to \infty$, the fraction of problem instances representable with $|T| \leq \alpha k$ converges to zero.*
2. *If $\frac{n}{k} \to 0$ as $k, n \to \infty$, the fraction of problem instances representable with $|T| \leq \alpha n$ converges to zero.*

For example, if we randomly select fifty sixteen-bit integers (so that trivially $6 \leq |T| \leq 16$), the probability of being able to represent this set with $|T| \leq 11$ is less than one in seventeen million.

5 A Greedy Heuristic

We propose a greedy heuristic for constructing a representing set T. The idea is that at each step, we choose a t which will enable us to represent the largest number of $s \in S$ which have not been represented already.

Start with $T = \emptyset$. In the first iteration, find the most common difference d in D_S and set $T_1 = \{d\}$. For all s_j such that $s_j - s_i = d$ for some i, remove s_j from S. The idea is that after we have removed s_j, the representation of s_j will be d plus the representation we eventually find for s_i. We say that s_j has been "retracted" into s_i. Let R_i be the representation of s_i, i.e. the subset of T such that $s_i = \sum_{t \in R_i} t$. So at this point we have the partial representation $R_j = \{d\} \cup R_i$. For notational convenience we let $s_1 = 0$, $R_1 = \emptyset$. When s_j is retracted into zero, or into an s_i whose representation was completed in a previous iteration, its representation is complete (as is the representation of anything that depends, directly or indirectly, upon R_j).

One additional detail is that we might not be able to simultaneously remove *all* such s_j, we can only remove s_j if we do not remove s_i (otherwise we would need two copies of d in T). This situation arises if d appears twice consecutively as a difference, $s_j = s_i + d$ and $s_k = s_j + d$. In such a sequence of repeated adjacent differences we can only take alternating elements out of S. We take this into account in selecting the best d.

In subsequent iterations we do not only look at differences among the remaining elements of S, but we also look at how the elements already in T can be used to build representations. We can have

$$s_j = s_i - t + t' + d \tag{3}$$

for any $t \in R_i$ and any $t' \in T - R_i$ such that

- there is no s_k whose current representation includes both t' and R_j (otherwise t' would occur twice in R_k)
- there is no s_k whose current representation uses R_j and intersects $R_i - \{t\}$ (otherwise elements of $R_k \cap (R_i - \{t\})$ would occur twice in R_k)

With proper bookkeeping we can enumerate all allowable choices of (i, j, t, t') for which $d > 0$ in (3). We also do not consider retraction into any s_i that has a partial representation: s_i must either have a completed representation, or not yet have any representation. Retraction into partially-represented elements would require additional bookkeeping and more complex restrictions on t and t'. Having enumerated all possibilities we add d^* to T, where d^* is the value which enables us to retract the largest number of s_j.

Of course we are not limited to a single t or t', we could consider arbitrary subsets of R_i and of $T - R_i$. But this leads to exponential growth in the number

n	$\lceil \log s_n \rceil$	Avg. Heuristic	pct. below trivial
8	6	5.42	57
10	6	5.76	24
15	6	6.0	0
10	8	6.98	90
15	8	7.83	16
20	8	7.98	2
15	12	10.42	93
20	12	11.38	59
30	12	12.0	0
25	16	15.28	64
30	16	15.91	9
40	16	16.0	0

Table 1: Results for instances uniformly randomly generated between 1 and $2^m - 1$; 100 trials each.

n	$\lceil \log s_n \rceil$	Avg. Heuristic	pct. below trivial
15	12	9.52	100
20	12	10.09	99
30	12	11.06	69
25	16	14.85	79
30	16	15.69	28
40	16	15.99	1

Table 2: Results for instances generated by random subsets of $\{1, 2, 8, 32, 64, 256, 1024, 2048\}$ (12-bit s_n) and $\{1, 4, 8, 16, 64, 128, 512, 1024, 2048, 8192, 16384, 32768\}$ (16-bit s_n); 100 trials each.

of options to consider. To maintain a polynomial running time we must limit the size of the subsets by a constant; the implementation used for the results of the next section considers pairs of subsets with union of size at most 3. Thus we may consider representations such as $s_j = s_i - t_1 - t_2 - t_3 + d$ or $s_j = s_i - t + t'_1 + t'_2 + d$ where $t_i \in R_i$ and $t'_i \in T - R_i$ satisfy the conditions above.

5.1 Experimental Results

We uniformly randomly generated sets of n m-bit numbers for n, m as shown in Table 1, computing the average size (over 100 random instances) of the representation found by the heuristic and the fraction of instances for which a better-than-trivial solution was obtained.

These results in Table 1 are difficult to interpret since we do not know what fraction of instances should have better-than-trivial solutions. We generated instances known to have nontrivial solutions by taking sums of random subsets of a set of generators of size less than $\log s_n$. In this case it is clear that the heuristic is rarely finding an optimal solution (which would be no larger than the known generating set), but it tends to find smaller generating sets than were found for uniformly generated instances. In no case did we find a generating set smaller than the set actually used to create the instance.

Acknowledgments

We would like to thank Shuang Luan, Cris Moore, and MohammadReza Salavatipour for useful discussions.

References

[1] N. Bansal, D. Coppersmith, and B. Schieber. Minimizing setup and beam-on times in radiation therapy. In *Proceedings of APPROX 2006*, 2006.

[2] M. Develin. Optimal subset representations of integer sets. *Journal of Number Theory*, 89:212–221, 2001.

[3] T. Kalinowski. The algorithmic complexity of the minimization of the number of segments in multileaf collimator field segmentation. preprint.

[4] S. Luan, D.Z. Chen, X.S. Hu, S.A. Naqvi, C. Wang, and C.X. Yu. Generalized geometric approaches for leaf sequencing problems in radiation therapy. *International Journal of Computational Geometry and Applications (IJCGA)*, 16:175–204, 2006.

[5] D. Mills. Some observations on subset sum representations. preprint.

[6] D. Moulton and D. Petrie. Representing powers of numbers as subset sums of small sets. *Journal of Number Theory*, 89:193–211, 2001.

[7] T. Schaefer. The complexity of satisfiability problems. In *Proc. 10th ACM Symp. on Theory of Computing*, pages 216–226, 1978.