Mechanical Proving with Walnut for Squares and Cubes in Partial Words

John Machacek
Department of Mathematics, University of Oregon, Eugene, OR, USA

Abstract

Walnut is a software that can prove theorems in combinatorics on words about automatic sequences. We are able to apply this software to both prove new results as well as reprove some old results on avoiding squares and cubes in partial words. We also define the notion of an antisquare in a partial word and begin the study of binary partial words which contain only a fixed number of distinct squares and antisquares.

2012 ACM Subject Classification Mathematics of computing → Combinatorics on words

Keywords and phrases Partial words, squares, antisquares, cubes, Walnut

Digital Object Identifier 10.4230/LIPIcs.CPM.2022.5

Funding NSF DMS-2039316.

1 Introduction

Our focus is on repetitions in partial words and the use of the software called Walnut[1] [15] to give automated proofs in situations where automatic sequences can be used. Partial words are generalizations of usual words that make use of an additional "wildcard" character which matches all other characters. Walnut has been used to give alternative proofs of previously known results and has also been used to produce new theorems in combinatorics on words (see e.g., [16]). To our knowledge this work is the first use of Walnut with partial words and contains proofs of both new results as well as previously known ones. We will define the notions which are most central to our work, but familiarity with some standard terms and ideas from combinatorics on words [13] is assumed.

A *square* or *cube* is a word of the form xx or xxx, respectively, for a nonempty word x. For example, $(010)^2 = 010010$ is a square and $(110)^3 = 110110110$ is a cube. We will consider words that avoid squares as well as those which avoid cubes. This means words that do not have a factor (i.e., contiguous substring) which is a square or cube respectively. A *squarefree* word is a word which avoids squares and a *cubefree* word is a word which avoid cubes. A *morphism* is a map $\psi : \Sigma^* \to \Delta^*$ between words over two alphabets such that $\psi(xy) = \psi(x)\psi(y)$ for all $x, y \in \Sigma^*$. We will make frequent use of morphisms to find words with a desired property.

A classic problem in combinatorics on words is constructing words avoiding squares, cubes, and other types of repetitions. Thue [20, 2] was able to construct an infinite cubefree binary word and an infinite squarefree ternary word each of which can be obtained as the fixed point of a morphism. We let

$$\mathbf{tm} = 01101001100101101001011001101001\cdots$$

denote the *Thue-Morse* word which is the fixed point of the morphism $0 \mapsto 01$ and $1 \mapsto 10$ which begins with 0. We also let

$$\mathbf{vtm} = 01202101210201202102012 1\cdots$$

[1] We have used the version of Walnut available at `https://github.com/DistortedLight/Walnut`.

denote the fixed point of the morphism $0 \mapsto 012$, $1 \mapsto 02$, and $2 \mapsto 1$ which is sometimes called the *ternary Thue-Morse word* (see e.g., [19]) or a *variant of the Thue-Morse word* (see [5]). It is the case that **tm** is cubefree and **vtm** is squarefree. We will make use of both of these words in constructions later.

A *partial word* is a word which can use a special character $\diamond$ which is called a *hole* or *wildcard*. For partial words a square or cube is a partial word $\mathbf{w}$ which is *contained* in a square $\mathbf{u} = xx$ or a cube $\mathbf{u} = xxx$ respectively in the sense the $\mathbf{w}[i] = \mathbf{u}[i]$ whenever $\mathbf{w}[i] \neq \diamond$. In this case we write $\mathbf{w} \subset \mathbf{u}$. The *order* of the square or cube is the length of x, which we denote by $|x|$. For example, $01101 \diamond 011$ is a partial word which is a cube of order 3 since it is contained in $(011)^3 = 011011011$. It is clear the presence of holes makes it more difficult to avoid squares or cubes. All words are partial words with no holes. When we wish to emphasize that a (partial) word has no holes we will refer to the word as a *full word*.

2 Squares, antisquares, cubes, and first-order logic

In this section we define what squares, antisquares, and cubes are in terms of first-order logic. This allows for seamless use with Walnut and highlights some differences between full words and partial words. All variables we quantify over are taken from the nonnegative integers unless specified otherwise.

For a full word $\mathbf{w}$ containing a *square* means

$$\exists j \exists (n > 0) \forall i, (i < n) \implies (\mathbf{w}[j+i] = \mathbf{w}[j+n+i])$$

while for a partial word it means

$$\exists j \exists (n > 0) \forall i, (i < n) \implies ((\mathbf{w}[j+i] = \mathbf{w}[j+n+i]) \vee (\mathbf{w}[j+i] = \diamond) \vee (\mathbf{w}[j+n+i] = \diamond))$$

both of which can be expressed in first-order logic. We see that the expression for partial words contains more clauses. A value of n for which the above is made true is called the *order* of the square.

A partial word $\mathbf{w}$ contains an *antisquare* provided

$$\exists j \exists (n > 0) \forall i, (i < n) \implies ((\mathbf{w}[j+i] \neq \mathbf{w}[j+n+i]) \wedge (\mathbf{w}[j+i] \neq \diamond) \wedge (\mathbf{w}[j+n+i] \neq \diamond))$$

which is consistent with what was considered for binary words in [17] and differs from the notion of an *anti-power* studied in [10]. We believe this to be a natural definition of an antisquare for a partial word given how it comes from negating the latter half of the implication in the logical expression for a square. We see that replacing a letter with a hole can create a square, and dually it can remove the presence of an antisquare. A factor which is an antisquare cannot contain any holes, but since we will consider squares and antisquares together in partial words holes will still play a crucial role.

Table 1 Logical operators and corresponding symbols in Walnut.

$\forall$	$\exists$	$\wedge$	$\vee$	$\neg$	$\Longrightarrow$
A	E	&	|	~	=>

Lastly, we say a partial word $\mathbf{w}$ contains a *cube* if

$$\begin{aligned}\exists j \exists (n>0) \forall i, (i<n) \implies \Big(& ((\mathbf{w}[j+i] = \mathbf{w}[j+n+i]) \wedge (\mathbf{w}[j+n+i] = \mathbf{w}[j+2n+i])) \\ & \vee ((\mathbf{w}[j+i] = \diamond) \wedge (\mathbf{w}[j+n+i] = \mathbf{w}[j+2n+i])) \\ & \vee ((\mathbf{w}[j+n+i] = \diamond) \wedge (\mathbf{w}[j+i] = \mathbf{w}[j+2n+i])) \\ & \vee ((\mathbf{w}[j+2n+i] = \diamond) \wedge (\mathbf{w}[j+i] = \mathbf{w}[j+n+i])) \\ & \vee ((\mathbf{w}[j+i] = \diamond) \wedge (\mathbf{w}[j+n+i] = \diamond)) \\ & \vee ((\mathbf{w}[j+i] = \diamond) \wedge (\mathbf{w}[j+2n+i] = \diamond)) \\ & \vee ((\mathbf{w}[j+n+i] = \diamond) \wedge (\mathbf{w}[j+2n+i] = \diamond)) \Big)\end{aligned}$$

where we find a more drastic difference compared to what would be the first-order logic for the case of full words. We note that in the logical expression for a cube we would only need

$$(\mathbf{w}[j+i] = \mathbf{w}[j+n+i]) \wedge (\mathbf{w}[j+n+i] = \mathbf{w}[j+2n+i])$$

in the latter half of the implication for full words. One could continue to consider higher powers, and the number of additional clauses in the partial word version will continue to grow. We will restrict our attention to squares and cubes.

The proofs of many results in this paper are given by short snippets of Walnut code. We now explain a few aspects of Walnut to make these snippets more readable to a reader that does not have prior experience with this language. A more detailed explanation can be found in [15]. One can see in Table 1 how usual logical symbols are represented in Walnut. Our alphabet will always be $\{0, 1, \ldots, N\}$ for some N, and we will use $N+1$ to denote the hole $\diamond$. For example, the binary partial word $0 \diamond 10$ in Walnut would be `0210`. The symbol `@` is use in denote a character of the alphabet as oppose of an integer which can be the index of a position in a word. So, `W[2] = @3` is used to say that the character 3 is in position 2 of the word W. Lastly, morphisms can be defined intuitively where `0->010` encodes $0 \mapsto 010$ the image of 0.

3 Results

3.1 Avoiding long squares in binary

In this subsection we look at binary partial words which only contain short squares and antisquares. It is not possible to completely avoid squares since any binary (full) word of length at least 4 will contain some square. Let the morphism $h : \{0, 1, 2\} \to \{0, 1, \diamond\}$ be defined by

$h(0) = 1100$
$h(1) = 011\diamond$
$h(2) = 1010$

which is a partial word variant of a morphism from [11, Section 2] that itself is a variant of a morphism used in [9] to produce a binary word which has no squares of order 3 or more. In particular, if the hole in the definition of the morphism is replaced with a 1, then the image under h of any ternary square-free word will be a binary word where all squares have order less than 3. With the addition of the hole we no longer avoid squares of order 3, but the resulting partial word will avoid squares of order 4 or more.

▶ **Theorem 1.** *The partial word $h(\mathbf{vtm})$ is a binary partial word with infinitely many holes that avoids squares of order 4 or more.*

Proof. The word **vtm** is in Walnut as `VTM`. So, all we need to do is define the morphism h and apply it to **vtm** and check for squares of order 4 or more. In the code below `Wh` denotes the image of this morphism. Recall, Walnut only uses $0, 1, \ldots,$ as letters. So, the image of h is a binary partial word represented over $\{0, 1, 2\}$ where 2 plays the role of $\diamond$. Running the following in Walnut

```
morphism h "0->1100, 1->0112, 2->1010";
image Wh h VTM;
eval no_sq "?msd_2 ~ Ej En Ai (n>3) & ((i<n)=>((Wh[j+i]=Wh[j+n+i])
        | Wh[i+j]=@2 | Wh[i+n+j]=@2))";
```

returns "TRUE" which proves the result. ◀

▶ Remark 2. To our knowledge the construction in Theorem 1 is new in the context of partial words, but the result on avoiding long squares is not optimal. In [4, Theorem 4] an infinite binary partial word with infinitely many holes is constructed so that the only squares compatible with factors of it are 0^2, 1^2, $(01)^2$, and $(11)^2$. This is based on a construction for full words from [18]. We have been unable to automate a version of this construction in Walnut due to the space required for the computation.

We now consider the morphism $f : \{0, 1, \ldots, 7\}^* \to \{0, 1, \ldots, 7\}^*$ given by

$$f(0) = 01 \qquad f(1) = 23$$
$$f(2) = 24 \qquad f(3) = 51$$
$$f(4) = 06 \qquad f(5) = 01$$
$$f(6) = 74 \qquad f(7) = 24$$

along with the coding $g : \{0, 1, \ldots, 7\}^* \to \{0, 1, \diamond\}$ by $g(m) = m \pmod 2$ for $m \neq 6$ and $g(6) = \diamond$. Applying g to the fixed point of f will give us a word avoiding both squares and antisquares of large length. Since $f(0) = 01$ we may iterate applying f to 0 to obtain the unique fixed point of the morphism f we denote by $f^\omega(0)$. We will make use of the notation f^ω to denote fixed points of morphisms elsewhere as well.

▶ **Theorem 3.** *The partial word $g(f^\omega(0))$ is a binary partial word with infinitely many holes that avoids squares of order 7 or more and avoids antisquares of order 3 or more.*

Proof. We establish the theorem by running the following in Walnut

```
morphism f "0->01, 1->23, 2->24, 3->51, 4->06, 5->01, 6->74, 7->24";
morphism g "0->0, 1->1, 2->0, 3->1, 4->0, 5->1, 6->2, 7->1";
promote Wf f;
```

Table 2 The length of the longest binary partial word with a single hole that contains at most a squares and at most b antisquares.

a \ b	0	1	2	3	4	5	6	7	8	9	10	11	12	13	⋯
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	⋯
1	3	4	5	5	5	5	5	5	5	5	5	5	5	5	⋯
2	5	7	9	10	11	11	12	12	14	14	15	16	16	16	⋯
3	7	11	14	19	19	19	19	22	26	30	34	52	97		
4	9	15	22	27	30	45	54	103	397						
5	11	19	35	40	74										
6	13	23	47	50											
7	15	27	59												
8	17	31	147												
9	19	35													
10	21	39													
⋮	⋮	⋮													

```
image Wg g Wf;
eval no_sq "?msd_2 ~ Ej En Ai (n>6) & ((i<n)=>((Wg[j+i]=Wg[j+n+i])
        | Wg[i+j]=@2 | Wg[i+n+j]=@2))";

eval no_anti "?msd_2 ~ Ej En Ai (n>2) & ((i<n)=>((Wg[j+i]!=Wg[j+n+i])
        & Wg[i+j]!=@2 & Wg[i+n+j]!=@2))";
```

which outputs “TRUE” twice. ◀

▶ Remark 4. In was shown in [17, Theorem 9] that the full word obtained by coding the fixing point of f with $m \mapsto m \pmod 2$ for all $m \in \{0, 1, \ldots, 7\}$ avoids both squares and antisquares of order 3 or more. Since adding holes cannot make any antisquares the fact about avoiding antisquares in Theorem 3 is immediate.

In [17] for any fixed a and b, the problem of finding the longest binary full word which contains at most a distinct squares and at most b distinct antisquares was solved. One can consider versions of the same problem for partial words. For example, we can ask for the length of the longest binary partial word with a fixed number of holes that contains at most a distinct squares and at most b distinct antisquares. We will focus on the case of partial words with a single hole. Counting distinct squares has received much attention for both words and partial words. It is known that even the addition of a single hole can fundamentally change distinct squares [6, 7, 12, 14]. Unlike if we were simply avoiding squares, the length of this longest binary partial word can be shorter or longer than that of the corresponding full word. This is since replacing a letter in a binary full word can possibly create a square while it also has the potential to remove an antisquare.

Consider the following example for $a = 4$ and $b = 5$. The length 32 partial word

$$\diamond 01110100000110100001100000010000$$

contains only the squares 0^2, 1^2, $(00)^2$, and $(10)^2$. It also contains only the antisquares 01, 10, 0011, 0110, and 1100. For full words the optimal length is 31 which was originally computed in [17, Figure 1]. One binary full word giving witness to this length is

$$0111010000011010000110000010000$$

which is obtained from the partial word above by removing the hole. Notice prepending 0 to this full word creates the new antisquare 001110 while prepending 1 creates 1011101000. Prepending $\diamond$ allows us to avoid new antisquares. Moreover, we can create even longer partial words with one hole and only 4 distinct squares and 5 distinct antisquares. The optimal length is 45. The partial word

$$000010000011000010110000011 \diamond 00101100000101110$$

has length 45 and only contains the squares 0^2, 1^2, $(00)^2$, and $(01)^2$ as well as only the antisquares 01, 10, 0011, 0110, and 1100. In Table 2 we list the optimal lengths for many values of a and b.

▶ **Theorem 5.** *The values in Table 2 are correct.*

Proof. The first three rows and first two columns are each infinite sequences of finite values and must be proven. Outside of these rows and columns there are only finitely many entries which can be computed. The first three rows follow from the fact that a binary partial word with one hole that contains at most 0, 1, or 2 squares has length at most 1, 5, or 16 respectively. This can be seen by direct verification of this fact, then computing the values until we reach 1, 5 or 16. Note it is clear the entries weakly increase along row and column.

For the first column, up to complementation, we consider words of the form $0^m \diamond 0^n$ or $0^m \diamond 1^n$ which are the only binary partial words with a single hole that do not contain an antisquare. Let $\ell = m + n + 1$ be the length of such a word. The number of squares such a word contains is

$$\left\lfloor \frac{m+1}{2} \right\rfloor + \left\lfloor \frac{n+1}{2} \right\rfloor = \left\lfloor \frac{\ell}{2} \right\rfloor$$

and so $\ell = 2a+1$ is the longest length containing at most a distinct squares and 0 antisquares.

Now for the second column we consider partial words with a single hole and only 1 antisquare. Let us assume $a > 1$. We will look at partial words which start with 0 and contain only the antisquare 01. So, we have $0^m 1^n \diamond 0^p 1^q$ with $m > 0$. Note if $m > 1$, then $n \leq 1$ or else we have both the antisquares 01 and 0011. Similarly if $p > 1$, then $q \leq 1$. So, let us assume our partial word is $0^m 1 \diamond 0^p 1$. This partial word has $\lfloor \frac{N}{2} \rfloor + 1$ distinct squares where $N = \max(m, p+1)$ unless $m = p+1$ and the whole partial word then gives an additional square. The squares contained are 1^2 and 0^{2k} for $2k < N$ along with possibly $(0^m 1)^2$. So, we may take $0^{2a-2}1 \diamond 0^{2a-2}1$ which has length $4a-1$ and contains a squares and 1 antisquare. This gives us one such word realizing the maximum agrees with what is found in Table 2. We also have the partial word $0^{2a-1}1 \diamond 0^{2a-2}$ of length $4a-1$ which contains a squares and 1 antisquare. It turns out that all other such partial words can be obtained from the two we have given by complement and reversal. This can be checked by considering the remaining cases of the possible forms of the partial word in a similar manner. ◀

▶ Remark 6. Table 2 is incomplete, and we do not know if the missing entries are finite or infinite. From [17] it is known that for full binary words the entry corresponding to $a = 5$ and $b = 5$ is finite while the remaining missing entries are infinite.

3.2 Avoiding non-trivial squares and cube with many holes

In this subsection we demonstrate how some known constructions [8] of partial words "dense" with holes avoiding powers can be obtained and verified through Walnut. The *hole sparsity* of a partial word is smallest s such that every factor of length s contains at least one hole. A

square of the form $a\diamond$ or $\diamond a$ for some letter a is called a *trivial square*. Indeed every partial word of length at least 2 which contains at least one hole as well as at least one letter will contain a trivial square. Thus for avoidance purposes we must allow trivial squares and avoid non-trivial squares. Since the presence of holes makes squares or cubes more likely, it is an interesting problem to find partial words with small hole sparsity (and hence many holes) which avoid non-trivial squares or cubes (or more generally higher powers).

Let us consider the morphism $\rho : \{0,1,2,3\}^* \to \{0,1,2,3\}^*$ defined by

$$\rho(0) = 03$$
$$\rho(1) = 12$$
$$\rho(2) = 01$$
$$\rho(3) = 10$$

which appears in [1, Exercise 33(c)]. Next we consider the morphism $\sigma : \{0,1,2,3\}^* \to \{0,1,2,3,\diamond\}^*$ defined by

$$\sigma(0) = 320\diamond$$
$$\sigma(1) = 120\diamond$$
$$\sigma(2) = 310\diamond$$
$$\sigma(3) = 130\diamond$$

whose image is a partial word over an alphabet with size 4. We are now ready to give on automated proof of the following which is in the proof of [8, Lemma 2].

▶ **Theorem 7.** *The partial word $\sigma(\rho^\omega(0))$ is a partial word with hole sparsity 4 over an alphabet of size 4 which avoids non-trivial squares.*

Proof. It is easy to see that the squares $\diamond^2$, 0^2, 1^2, 2^2, and 3^2 do not occur in $\sigma(\rho^\omega(0))$. Recall, since we avoiding non-trivial squares $a\diamond$ and $\diamond a$ are allowed to we present for $a \in \{0,1,2,3\}$. Thus, we only need to worry about squares of order n for $n > 1$. Running the following commands in Walnut

```
morphism rho "0->03, 1->12, 2->01, 3->10";
morphism sigma "0->3204, 1->1204, 2->3104, 3->1304";
promote Wrho rho;
image W sigma Wrho;
eval no_sq "?msd_2 ~ Ej En Ai (n>1) & ((i<n)=>((W[j+i]=W[j+n+i])
        | W[i+j]=@4 | W[i+n+j]=@4))";
```

results in an output of "TRUE" and the theorem is proven. ◀

We let τ denote the morphism defined by $\tau(0) = 01\diamond$ and $\tau(1) = 02\diamond$. We can now give an automated proof of the following which was first proven in [8, Lemma 7].

▶ **Theorem 8.** *The partial word $\tau(\mathbf{tm})$ is partial word with hole sparsity 3 over an alphabet of size 3 which avoids cubes.*

Proof. The word $\mathbf{tm}$ is contained in Walnut at `T`. We run the following in Walnut

```
morphism tau "0->013, 1->023";
image W tau T;
eval no_cube "?msd_2 ~Ej En Ai (n>1)&((i<n)=>(
```

```
W[j+i]=W[j+n+i] & W[j+n+i]=W[j+2*n+i])
|(W[j+i]=@3 & W[j+n+i]=W[j+2*n+i])|(W[j+n+i]=@3 & W[j+i]=W[j+2*n+i])
|(W[j+2*n+i]=@3 & W[j+i]=W[j+n+i])|(W[j+i]=@3 & W[j+n+i]=@3)
|(W[j+i]=@3 & W[j+2*n+i]=@3) | (W[j+n+i]=@3 & W[j+2*n+i]=@3))";
```

and it outputs "TRUE" proving the theorem. ◀

▶ Remark 9. Both Theorem 7 and Theorem 8 are optimal in establishing smallest hole sparsity avoiding non-trivial squares and cubes respectively for a given alphabet size [3]. For example, there does not exists an infinite partial word with hole sparsity 3 over a 4 letter alphabet which avoids non-trivial squares.

4 Conclusion

We have initiated a study of partial words paired with the theorem prover Walnut. Additionally, we have extended the definition of antisquares to partial words. We believe both directions could be a source of new problems in combinatorics on words. Furthermore, we have given alternative proofs of some results on partial words which provide machine verification. We have discussed how the logical statements expressing a square are longer for partial words than full words. So this adds some complexity to the Walnut calculations. Additionally, in Walnut partial words work with an alphabet with an extra letter which represents the hole. Let us close with an example comparing partial words with full words in Walnut.

Let us consider the morphisms g and h defined by

$$g(0) = 1100 \qquad h(0) = 1100$$
$$g(1) = 0111 \qquad h(1) = 011\diamond$$
$$g(2) = 1010 \qquad h(2) = 1010$$

where h was the morphism used in Theorem 1. To get an idea of what happens going from full words to partial words we have the deterministic finite automata with output (DFAO) for $g(\mathbf{vtm})$ and $h(\mathbf{vtm})$ shown in Figure 1 and Figure 2 respectively. Walnut works with automatic sequences using their DFAOs. We find in this case only a modest increase in the size of the DFAO, and we were indeed able to use Walnut to automate a proof in the more complex but still tractable partial word situation.

To show there are no squares of length greater than 3 in neither $g(\mathbf{vtm})$ nor $h(\mathbf{vtm})$ we may run

```
morphism h "0->1100, 1->0112, 2->1010";
image Wh h VTM;
eval no_sq "?msd_2 ~ Ej En Ai (n>3) & ((i<n)=>((Wh[j+i]=Wh[j+n+i])
        | Wh[i+j]=@2 | Wh[i+n+j]=@2))";

morphism g "0->1100, 1->0111, 2->1010";
image Wg g VTM;
eval no_sq_full "?msd_2 ~ Ej En Ai (n>3) & ((i<n)=>(Wg[j+i]=Wg[j+n+i]))";
```

in Walnut. In the log files produced we will find the following two lines

```
(i<n=>((Wh[(j+i)]=Wh[((j+n)+i)]|Wh[(i+j)]=@2)|Wh[((i+n)+j)]=@2)):229 states - 24ms
```

and

```
(i<n=>Wg[(j+i)]=Wg[((j+n)+i)]):217 states - 10ms
```

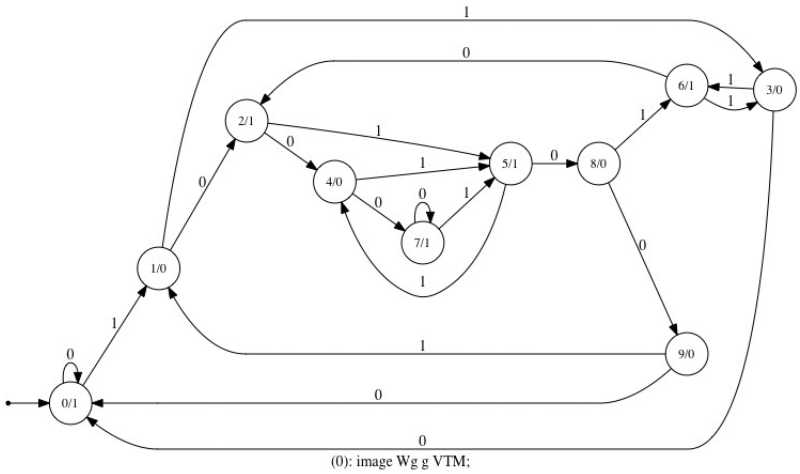

Figure 1 The DFAO for $g(\mathbf{vtm})$.

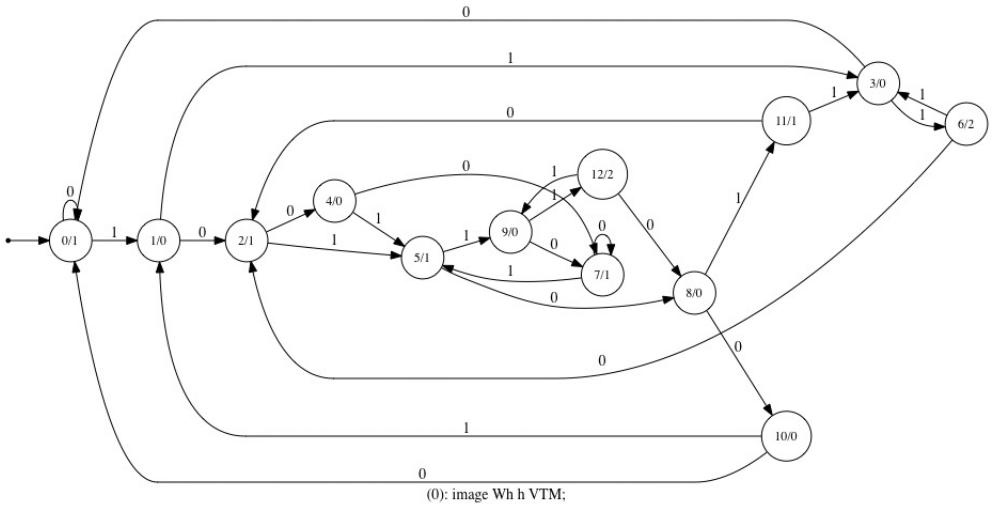

Figure 2 The DFAO for $h(\mathbf{vtm})$.

showing the sizes of the automata Walnut needs to determine the exists of a square in either $g(\mathbf{vtm})$ or $h(\mathbf{vtm})$ respectively. We see the partial word version requires 229 states more states and takes to build 24ms compared 217 states and to 10ms for the full word version. The issue one encounters in Walnut computations is typically in issue of space due to building some automaton. This example, and the others we have given, suggest that partial word versions of theorems may take slightly more space in Walnut but may often tractable with Walnut when their full word counterparts are.

References

1 Jean-Paul Allouche and Jeffrey Shallit. *Automatic sequences*. Cambridge University Press, Cambridge, 2003. Theory, applications, generalizations. doi:10.1017/CBO9780511546563.

2 Jean Berstel. Axel thue's papers on repetitions in words: a translation. Publications de Laboratoire de Combinatoire et d'Informatique Mathématique 20, Université du Québec á Montréal, 1995.

3 Kevin Black, Francine Blanchet-Sadri, Ian Coley, Brent Woodhouse, and Andrew Zemke. Pattern avoidance in partial words dense with holes. *J. Autom. Lang. Comb.*, 22(4):209–241, 2017. doi:10.25596/jalc-2017-209.

4 F. Blanchet-Sadri, Ilkyoo Choi, and Robert Mercaş. Avoiding large squares in partial words. *Theoret. Comput. Sci.*, 412(29):3752–3758, 2011. doi:10.1016/j.tcs.2011.04.009.

5 F. Blanchet-Sadri, James D. Currie, Narad Rampersad, and Nathan Fox. Abelian complexity of fixed point of morphism $0 \mapsto 012$, $1 \mapsto 02$, $2 \mapsto 1$. *Integers*, 14:Paper No. A11, 17, 2014.

6 F. Blanchet-Sadri, Yang Jiao, John M. Machacek, J. D. Quigley, and Xufan Zhang. Squares in partial words. *Theoret. Comput. Sci.*, 530:42–57, 2014. doi:10.1016/j.tcs.2014.02.023.

7 F. Blanchet-Sadri, Robert Mercaş, and Geoffrey Scott. Counting distinct squares in partial words. *Acta Cybernet.*, 19(2):465–477, 2009.

8 Francine Blanchet-Sadri, Kevin Black, and Andrew Zemke. Unary pattern avoidance in partial words dense with holes. In Adrian-Horia Dediu, Shunsuke Inenaga, and Carlos Martín-Vide, editors, *Language and Automata Theory and Applications - 5th International Conference, LATA 2011, Tarragona, Spain, May 26-31, 2011. Proceedings*, volume 6638 of *Lecture Notes in Computer Science*, pages 155–166. Springer, 2011. doi:10.1007/978-3-642-21254-3_11.

9 R. C. Entringer, D. E. Jackson, and J. A. Schatz. On nonrepetitive sequences. *J. Combinatorial Theory Ser. A*, 16:159–164, 1974. doi:10.1016/0097-3165(74)90041-7.

10 Gabriele Fici, Antonio Restivo, Manuel Silva, and Luca Q. Zamboni. Anti-powers in infinite words. *J. Combin. Theory Ser. A*, 157:109–119, 2018. doi:10.1016/j.jcta.2018.02.009.

11 Daniel Gabric and Jeffrey Shallit. The simplest binary word with only three squares. *RAIRO Theor. Inform. Appl.*, 55:Paper No. 3, 7, 2021. doi:10.1051/ita/2021001.

12 Vesa Halava, Tero Harju, and Tomi Kärki. On the number of squares in partial words. *RAIRO Theor. Inform. Appl.*, 44(1):125–138, 2010. doi:10.1051/ita/2010008.

13 M. Lothaire. *Combinatorics on words*. Cambridge Mathematical Library. Cambridge University Press, Cambridge, 1997. doi:10.1017/CBO9780511566097.

14 John Machacek. Partial words with a unique position starting a square. *Inform. Process. Lett.*, 145:44–47, 2019. doi:10.1016/j.ipl.2019.01.010.

15 Hamoon Mousavi. Automatic theorem proving in Walnut. arXiv:1603.06017.

16 Hamoon Mousavi, Luke Schaeffer, and Jeffrey Shallit. Decision algorithms for Fibonacci-automatic words, I: Basic results. *RAIRO Theor. Inform. Appl.*, 50(1):39–66, 2016. doi:10.1051/ita/2016010.

17 Tim Ng, Pascal Ochem, Narad Rampersad, and Jeffrey Shallit. New results on pseudosquare avoidance. In *Combinatorics on words*, volume 11682 of *Lecture Notes in Comput. Sci.*, pages 264–274. Springer, Cham, 2019. doi:10.1007/978-3-030-28796-2_21.

18 Narad Rampersad, Jeffrey Shallit, and Ming-wei Wang. Avoiding large squares in infinite binary words. *Theoret. Comput. Sci.*, 339(1):19–34, 2005. doi:10.1016/j.tcs.2005.01.005.

19 Michaël Rao, Michel Rigo, and Pavel Salimov. Avoiding 2-binomial squares and cubes. *Theoret. Comput. Sci.*, 572:83–91, 2015. doi:10.1016/j.tcs.2015.01.029.

20 Axel Thue. *Selected mathematical papers*. Universitetsforlaget, Oslo, 1977. With an introduction by Carl Ludwig Siegel and a biography by Viggo Brun, Edited by Trygve Nagell, Atle Selberg, Sigmund Selberg, and Knut Thalberg.