Tag systems and Collatz-like functions

Liesbeth De Mol

Gent University, Centre for Logic and Philosophy of Science, Blandijnberg 2, 9000 Gent, Belgium.

Abstract

Tag systems were invented by Emil Leon Post and proven recursively unsolvable by Marvin Minsky. These production systems have shown very useful in constructing small universal (Turing complete) systems for several different classes of computational systems, including Turing machines, and are thus important instruments for studying limits or boundaries of solvability and unsolvability. Although there are some results on tag systems and their limits of solvability and unsolvability, there are hardly any that consider *both* the shift number v, as well as the number of symbols μ. This paper aims to contribute to research on limits of solvability and unsolvability for tag systems, taking into account these two parameters. The main result is the reduction of the $3n+1$-problem to a surprisingly small tag system. It indicates that the present unsolvability line – defined in terms of μ and v – for tag systems might be significantly decreased.

Key words: Tag Systems, limits of solvability and unsolvability, universality, $3n+1$-problem, Collatz-like functions

Email address: `elizabeth.demol@ugent.be` (Liesbeth De Mol).

1 Introduction

Already in 1921 Emil Leon Post proved the unsolvability of certain decision problems, rooted in what Martin Davis has called Post's thesis [7][8]. In 1943 a paper appeared by Post that summarizes the main results from this earlier research [34], but it was only in 1965 that Martin Davis posthumously published a manuscript by Post describing these earlier results in more detail [35]. As was argued in [27], basic to these results was Post's construction of tag systems. After nine months of research, trying to prove the recursive solvability of these systems, he concluded for a reversal of his entire program. He was now convinced that there might exist unsolvable decision problems in mathematics and logic. Since he never wanted to work on tag systems again, he never proved them unsolvable. In the end it was Marvin Minsky who proved that tag systems are indeed recursively unsolvable, as Post suspected, by proving that any Turing machine can be simulated by a tag system with a shift number $v = 6$ [23].

1.1 A short introduction to tag systems

A tag system consists of a finite alphabet $\Sigma = \{a_0, a_1, ..., a_{\mu-1}\}$ of μ symbols, a shift number $v \in \mathbb{N}$ and a finite set of μ words defined over the alphabet, including the empty word ϵ. Each of these words corresponds with one of the letters from the alphabet as follows:

$$
\begin{array}{ll}
a_0 & \rightarrow a_{0,1}a_{0,2}...a_{0,n_0} \\
a_1 & \rightarrow a_{1,1}a_{1,2}...a_{1,n_1} \\
... & ...\ ... \\
a_{\mu-1} & \rightarrow a_{\mu-1,1}a_{\mu-1,2}...a_{\mu-1,n_{\mu-1}}
\end{array}
$$

where each $a_{i,j} \in \Sigma, 0 \leq i < \mu$. Now, given an initial word A_0, the tag system tags the word associated with the leftmost letter of A_0 at the end of A_0, and deletes its first v symbols. This process is iterated until the tag system halts, i.e. produces a word A_i, after i iterations, having a length smaller than v. If this does not happen the tag system can become periodic or show divergent behaviour.
To give an example, consider the case where $v = 3$, $0 \rightarrow 00$, $1 \rightarrow 1101$, with

$A_0 = 10111011101000000$. We then get:

10111011101000000
110111010000001101
1110100000011011101
01000000110111011101
0000011011101110100
001101110111010000
10111011101000000

The word A_0 is reproduced after 6 steps, thus leading to the production of a period of length 6.
As simple as the definition of a tag system might be, they are very hard to study, and until now not very much is really known about these systems. Even the class of tag systems with $\mu = 2, v > 2$ is still not known to be recursively solvable or unsolvable. Indeed, the seemingly simple example mentioned above, first described by Post [35],[34], is still an open problem. Watanabe [40] studied this one specific case, trying to get a more formal grip on the periodic behaviour of this tag system. Although Watanabe's paper is very interesting, it contains rather fundamental errors. These lead to a wrong deduction of only three possible basic periodic structures for this tag system, while it can be proven that there are at least six. For more details, the reader is referred to [30]. Besides Watanabe, Minsky and Brian Hayes did some research on this one tag system (See for example [10], [11], [26]) again without any definite results concerning the solvability of this tag system.

1.2 Results on limits of solvability and unsolvability in tag systems

Despite the fact that tag systems have not been that well-studied as e.g. Turing machines, there are some significant results concerning their limits of solvability and unsolvability, i.e., results that help to determine largest recursively solvable and smallest universal (Turing complete) classes of tag systems.
In his posthumously published paper [35], Post mentions that the halting and reachability problem for the class of tag systems with $v = 1$ or $\mu = 1$ are trivially solvable. He furthermore notes that he completely solved these two decision problems for the case with $\mu = v = 2$, and considered this proof as the major result from his Procter fellowship in Princeton (1920–1921). The proofs however were never published. Wang [39] provided the proof for the case with

$v = 1$. We were able to find such a proof for the class $\mu = v = 2$, involving the application of a combinatorial kind of method applied to a rather large number of different subcases [28], [29]. Cocke and Minsky proved that any Turing machine can be simulated by a tag system for which $v = 2$ (See [2], [24], [25]), improving the result by Minsky [23]. Maslov generalized this result and proved that for any $v > 1$ there exists at least one tag system with an unsolvable decision problem and, independent of Wang, furthermore proved that any tag system for which $v = 1$ is recursively solvable [20].
Both μ and v can thus be regarded as *decidability criteria* [18] for tag systems, since their recursive solvability depends on the size of these parameters. Another such criterion for tag systems is the length of the words. Let l_{min} denote the length of the smallest word of a tag system and l_{max} the length of the lengthiest word. Wang proved that any tag system for which $l_{min} \geq v$ or $l_{max} \leq v$, has a solvable halting and reachability problem [39]. It should be added here that Maslov proved that the tag systems with an unsolvable decision problem that can be constructed using his method, for each $v > 1$ all satisfy the following condition: $l_{min} = v - 1$, $l_{max} = v + 1$ [20]. Taking into account Wang's result, he describes this condition as a kind of minimal condition for unsolvability in tag systems. This result was independently proven by Wang for a tag system with $v = 2$ [39].
As is clear from the previous, except by Post, the number of symbols μ of a tag system determining the number of words and thus production rules, has hardly been taken into account in the existing literature on tag systems. The role of μ however should not be underestimated. Its value not only determines the number of production rules for a given tag system, but also marks the difference between recursively solvable and unsolvable classes. In this respect, we would like to propose the following definition of a measure for the size of tag systems, including μ:

Definition 1.1 *The size of a tag system is defined as the product of μ and v, where $TS(\mu, v)$ denotes the class of tag systems with μ symbols and a shift-number v.*

The length of the words is not taken into account, since the decidability criterion with respect to l_{min} and l_{max} is defined relative to v.
Besides the existing results on the limits of solvability and unsolvability in tag systems, there are also several basic results that (directly or indirectly) use tag systems to construct smallest universal systems. For example, all the smallest known universal (Turing complete) Turing machines are (efficient) simulators of either tag systems [1], [14], [23], [37], [38] or bi-tag systems, a variant of tag systems [32], [33]. An up-to-date overview of the present situation of the boundaries of solvability and unsolvability in Turing machines can be found

in [32] (See Sec. 3 for more details).[1] Tag systems have also been used in the context of cellular automata. Matthew Cook's proof [5] of the fact that the cellular automaton rule 110 is weakly universal is indirectly based on the simulation of tag systems, through simulation of cyclic tag systems. (Weak universality here means that the cellular automaton starts from an infinite, ultimately periodic configuration. See for example [19].) On the basis of this result, he was able to construct very small weakly universal Turing machines. Another class of examples of small universal systems simulating tag systems are circular Post machines [13].
Given on the one hand, this significance of tag systems in the general research context of constructing small universal systems, and, on the other hand, their formal simplicity, it is considered interesting to study the boundaries of solvability and unsolvability in tag systems. However, because the number of symbols μ has hardly been taken into account since Post studied these systems, there are not much results in this context that consider *both* μ and v. As a consequence the smallest universal tag systems known are still quite large, leaving a huge gap between the known solvable classes of tag systems and the universal classes (See Sec. 3 for a more detailed discussion).
In the present paper (Sec. 2.1) we will show that the $3n+1$-problem can be reduced to a surprisingly small tag system from the class TS(3,2). This result shows that a proof of the recursive solvability of TS(3,2) depends on the famous $3n+1$-problem.[2] We will furthermore give an alternative proof of the recursive unsolvability of tag systems, by providing a method for reducing any Collatz-like function to a tag system (Sec. 2.2). In section 3 we will discuss the main result of this paper in the context of the boundaries of solvability and unsolvability in tag systems, as compared to similar results for Turing machines.

2 A simple, efficient encoding of Collatz-like functions in tag systems

Let $C : \mathbb{N} \to \mathbb{N}$ be defined by:

$$C(n) = \begin{cases} \frac{n}{2} & \text{if n is even} \\ 3n+1 & \text{if n is odd} \end{cases}$$

[1] The fact that the known small universal Turing machines simulating tag systems are efficient simulators of Turing machines, is due to Thurlough Neary and Damien Woods [31].
[2] It should be noted that Brian Hayes already mentioned the possible connection between the $3n+1$-problem and tag systems [10].

The $3n+1$-problem is the problem to determine for any $n \in \mathbb{N}$, whether $C(n)$ will end in a loop $C(4) = 2, C(2) = 1, C(1) = 4$, after a finite number of iterates.
The well-known $3n+1$-problem is one of those problems from number theory for which the statement of the problem is as simple as the problem is intractable. A survey on the $3n+1$ problem can be found in [15], [16], where [16] is a more recent and seriously extended version of [15]. An annotated bibliography can be found via Arxiv [17]. Although the general consensus is that $C(n)$ will always end in the same loop after a finite number of iterates for arbitrary n, no proof has yet been found. A nice illustration of the difficulties involved with the $3n+1$-problem is given by the following quote by Kakutani: [3]

> For about a month everybody at Yale worked on it, with no result. A similar phenomenon happened when I mentioned it at the University of Chicago. A joke was made that this problem was part of a conspiracy to slow down mathematical research in the U.S.

In [22] Pascal Michel considers generalized functions of C, called Collatz-like functions and proved that some of these functions can be reduced to Turing machines in between the known solvable and universal classes. These functions are based on the following equivalent form of the $3n+1$-function:

$$C'(2m) = m,$$
$$C'(2m+1) = 3m+2.$$

Given integers $d \geq 2; a_0, a_1, ..., a_{d-1}; r_0, r_1, ..., r_{d-1}; x \in \mathbb{N}$ a Collatz-like function is defined as follows: [4]

$$G(n) = \begin{cases} m_0 & \text{If } n \equiv 0 \bmod d \\ m_1 & \text{If } n \equiv 1 \bmod d \\ \vdots & \\ m_{d-1} & \text{If } n \equiv (d-1) \bmod d \end{cases}$$

where m_i is either undefined or denotes an operation of the following form:

$$\frac{a_i(n-i)}{d} + r_i$$

[3] Quoted in [16] from a private conversation dated 1981, Kakutani describing what happened after he circulated the problem around 1960

[4] It should be noted that Michel further extends these functions to functions of pairs of integers.

Similar generalizations were already considered by Conway [3] in 1972. He proved that these generalizations lead to Collatz-like problems which are generally unsolvable. I.e. he proved that there exists no method to decide whether a Collatz-like function G, when applied to a number n, will produce 1 after a finite number of iterates by proving that any register machine can be simulated by such a function. About 15 years later, Conway developed a simple universal programming language called Fractran [4], for doing arithmetic, its syntax being based on the methods he used in 1972. He furthermore constructed a universal fraction game, called the Polygame, on the basis of which one can rather easily construct a universal Collatz-like function. In [12], Kaščák gives an explicit construction of a universal *one-state linear operator algorithm*, involving a generalization of the Collatz-problem similar to Michel's, with a small modulus, equal to 396.

2.1 Reduction of the $3n + 1$-function in tag systems

In this section we will prove the following theorem:

Theorem 2.1 *The function $C(n)$ is reducible to a tag system T_C with $\mu = 3$, $v = 2$.*

Let A^i denote a string A repeated i times, $A \overset{\circ}{\to} B$ is the string B produced from A, after all the letters from A have been erased. Let $\Sigma = \{\alpha, c, y\}$ and $n \in \mathbb{N}$. Then, each iteration of $C(n)$ corresponds to the production of a string $\alpha^{C(n)}$ from a string α^n in T_C. The production rules are:

$$\begin{aligned} \alpha &\to cy \\ c &\to \alpha \\ y &\to \alpha\alpha\alpha \end{aligned}$$

Now, if n is of the form $2m$, T_C produces $\alpha^{\frac{n}{2}}$ from α^n:

$$\begin{aligned} \alpha^n &\overset{\circ}{\to} (cy)^{\frac{n}{2}} \\ (cy)^{\frac{n}{2}} &\overset{\circ}{\to} \alpha^{\frac{n}{2}} \end{aligned}$$

If n is of the form $2m+1$, T_C produces $\alpha^{3(\frac{n-1}{2})+2}$ $(= \alpha^{3m+2})$ from α^n:

$$\begin{aligned} \alpha^n &\overset{\circ}{\to} y(cy)^{\frac{n-1}{2}} \\ y(cy)^{\frac{n-1}{2}} &\overset{\circ}{\to} \alpha^{3(\frac{n-1}{2})+2} \end{aligned}$$

This encoding allows for efficient simulation of $C(n)$ for any n. If n is even, C_T needs n iterations, with n uneven, $n+1$, to simulate one iteration of $C(n)$. The reason for the simplicity of this encoding is that $C(n)$ relies on modulo operations, while tag systems themselves can be regarded as some kind of modulo systems. Indeed, the encoding is based on this one feature of tag systems. Consider a string A of length $|A|$, and let $A \xrightarrow{\circ} B$. Clearly, the length of B depends on $|A| \bmod v$, in that the "original" length of B (the addition of the lengths of the words produced from A) will be decreased with the additive complement of $|A| \bmod v$ (the additive complement of $b \bmod v$ is defined as $-b \bmod v$ evaluated to its least positive remainder, 0 included) In this respect, $|A| \bmod v$ determines what sequence of letters in B will and will not be scanned by the tag system. This feature is not only basic to our encoding, but is also the main ingredient in Minsky's and Cocke's proof of the universality of tag systems with $v = 2$ (See Sec. 1.1). To return to our encoding of C in T_C, if $|\alpha^n|$ is even, $|\alpha^n| \xrightarrow{\circ} (cy)^{\frac{n}{2}}$, with $|(cy)^{\frac{n}{2}}| \bmod v = 0$, guaranteeing that only the letter c will be scanned in B. Similarly, since $|(cy)^{\frac{n}{2}}|$ is even, no letter from $\alpha^{\frac{n}{2}}$ will have been erased after all the letters of $|(cy)^{\frac{n}{2}}|$ have been erased. In case $|\alpha^n|$ is uneven, $|\alpha^n| \xrightarrow{\circ} B$, with $|B| \bmod v = 1$, the first leading c being erased when the last α in α^n has been scanned. As a result, the tag system will scan the sequence of letters y. Although, taking together all the y's results in $\alpha^{3(\frac{n-1}{2})+3}$, the oddness of $y(cy)^{\frac{n-1}{2}}$ guarantees that the leading α will be erased after the last y has been scanned, thus leading to the desired result.
It should be noted here that T_C satisfies the minimal condition discussed by Maslov (Sec. 1.1). Indeed, $l_{min} = v - 1$ and $l_{max} = v + 1$.
Furthermore, the problem to decide for any n, whether C_n will ever lead to 1 after a finite number of iterations, reduces to the question of whether T_C will ever produce α. In other words, the $3n+1$-problem can be reduced to a reachability problem for T_C.

2.2 Generalization of the method to arbitrary Collatz-like functions

By generalizing and slightly changing the encoding from the previous section, we were able to prove the following theorem:

Theorem 2.2 *Given an arbitrary Collatz-like function $G(n)$, with modulus d. Then, there is always a tag system T_G with $v = d, \mu \leq 2d+3, \Sigma = \{h, \alpha, \alpha_0, \beta_0, \beta_1, ..., \beta_{d-1}, b_0, b_1, ..., b_{d-1}\}$ that simulates $G(n)$ for any n.*

Note that μ and v are completely determined by the modulus. The symbol h functions as a kind of halting symbol, used for those cases when $G(n)$, $n = dm + i$, $0 \leq i < d$, is undefined for i. It is also important to note that the encoding of the present section needs the extra symbols $\alpha_0, \beta_0, \beta_1, ..., \beta_{d-1}$.

Each iteration of G over a number n corresponds to the production of a string $\alpha_0\alpha^{G(n)}$ from a string $\alpha_0\alpha^n$. The production rules for α_0, α are:

$$\begin{aligned}\alpha_0 &\rightarrow \beta_{d-1}\beta_{d-2}...\beta_0\\ \alpha &\rightarrow b_{d-1}b_{d-2}...b_0\end{aligned}$$

If $G(n)$ is defined, with $n = dm + i$, $0 \leq i < d$, the production rules for β_i and b_i are :

$$\begin{aligned}\beta_i &\rightarrow (\alpha)^j\alpha_0(\alpha)^{r_i}\\ b_i &\rightarrow (\alpha)^{a_i}\end{aligned}$$

where j is the additive complement of $(i+1)$ relative to d [i.e. $: -(i+1) \bmod d$ evaluated to its least positive remainder], with $i = n \bmod d$.
If $G(n)$ is undefined, $n = dm + i$, $0 \leq i < d$, the production rules for β_i and b_i are:

$$\begin{aligned}\beta_i &\rightarrow h\\ b_i &\rightarrow h\end{aligned}$$

The production rule for h is:

$$h \rightarrow \epsilon$$

Now, applying the production rules of T_G to a given string $\alpha_0\alpha^n$, in case $G(n)$ is defined, we get:

$$\alpha_0\alpha^n \overset{\circ}{\rightarrow} \beta_i\beta_{i-1}...\beta_0(b_{d-1}b_{d-2}...b_0)^{\frac{n-i}{d}} \tag{1}$$

Note, that we again use the property, mentioned in Sec. 2.1, that the length of a string B produced from a string A, through $\overset{\circ}{\rightarrow}$, is completely determined through $|A| \bmod v$, i.e. if the additive complement c of $|A| \bmod v > 0$, then the first c letters of the first word(s) produced from A will be erased, when the last letter of A has been scanned. Note that it is because the number of letters erased is equal to c, that the order of the indices of the letters in the words produced from $\alpha_0, \alpha, \beta_i, b_i, 0 \leq i < d$ is reversed, thus being able to keep track of the remainder. Furthermore, by adding the extra symbol α_0, the rules assure that $b_{d-1}b_{d-2}...b_0$ will be repeated $m = \frac{n-i}{d}$ times.

After the application of one iteration on the string produced in (1), T_G produces:

$$b_i b_{i-1}...b_0(b_{d-1}b_{d-2}...b_0)^{\frac{n-i}{d}-1}(\alpha)^j\alpha_0(\alpha)^{r_i} \tag{2}$$

From (2), T_G produces

$$\underbrace{b_i b_{i-1}...b_0(\alpha)^j}_{d}\alpha_0(\alpha)^{a_i\left(\frac{n-i}{d}-1\right)+r_i} \tag{3}$$

after (n-i)/d - 1 iterations. As is clear, the symbol β_i produced through α_0 is used to assure the tag system will start scanning α_0 after one iteration of G has been completed, through the addition of j times α, since

$$i + 1 + j = d.$$

Furthermore, β_i is used to add r_i if $G(n)$ is defined and $r_i > 0$. The letter b_i is used to perform the multiplication of m with a_i, since b_i is repeated $m = (n - i)/d$ times.
From (3) T_G finally produces:

$$\alpha_0(\alpha)^{a_i\frac{n-i}{d}+r_i} \tag{4}$$

after one more iteration.
If we apply the production rules to a string $\alpha_0\alpha^n$, in the case $G(n)$ is undefined, the production given in (1) remains unchanged. Then

$$\beta_i\beta_{i-1}...\beta_0(b_{d-1}b_{d-2}...b_0)^{\frac{n-i}{d}} \overset{\circ}{\rightarrow} h^{\frac{n-1}{d}+1} \tag{5}$$

From (5) we finally get:

$$h^{\frac{n-1}{d}+1} \overset{\circ}{\rightarrow} \epsilon \tag{6}$$

leading the tag system to a halt.
As is clear, the encoding of Collatz-like functions into tag systems is very straightforward, the input n for G being directly encoded as a string of length $n+1$. As was the case for the reduction of the $3n+1$-problem, the simulation of Collatz-like functions is efficient, where one iteration of $G(n)$ maximally takes $2(\lfloor n/d \rfloor + 1)$ iterations in T_G.
Given the fact that Collatz-like functions are recursively unsolvable, as was

proven by Conway, the reduction of the present section serves as an alternative proof of the recursive unsolvability of tag systems. The unsolvable decision problem to determine for any Collatz-like function G and any integer n, whether it will ever produce the number 1 after a finite number of steps, reduces to the reachability problem to determine for any tag system T_G and any integer n whether it will ever produce the string $\alpha_0\alpha$ when started with initial condition $\alpha_0\alpha^n$.
In comparing the encoding of the present section with that from Sec. 2.1, it is clear that the encoding of the present section leads to the simulation of the $3n+1$-problem in a larger tag system, with $\mu = 6$. This is due to the use of the symbol α_0. One might thus wonder whether there is a condition under which a tag system T_G, encoding a function $G(n)$ using α_0, can be reduced to a smaller tag system T'_G, without α_0.[5] The following theorem gives such a condition as well as the production rules of T'_G, which are based on the encoding of the $3n+1$-problem from Sec. 2.1 in T_C.

Theorem 2.3 *Given a Collatz-like function $G(n)$ with modulus d, where for each n, $G(n)$ either undefined or equal to $\frac{a_i(n-i)}{d} + r_i$, $i = 0, 1, ..., d-1$. Then $G(n)$ can always be reduced to a tag system T'_G with $v = d, \mu \leq 2 + d, \Sigma = \{h, \alpha, b_0, b_1, ..., b_{d-1}\}$ iff. for every $G(n)$ defined, $\bar{i} < a_i$, if $i > 0$, $r_i = a_i - \bar{i}$, if $i = 0, r_i = 0$, where $\bar{i}$ is the additive complement of i. For each $G(n)$ defined, the production rules of T'_G are: $\alpha \to b_0 b_{d-1}..b_2 b_1; b_i \to \alpha^{a_i}$. For $G(n)$ undefined, the production rules are $b_i \to h$; $h \to \epsilon$*

The details of the proof are left to the reader.

3 Discussion: Collatz-like functions and limits of unsolvability.

It is a well-known fact that presently there exists a gap between classes of Turing machines that are known to be recursively solvable and classes which are not, because they have been proven universal (Turing complete). As was mentioned in Sec. 1.2, we are confronted with the same problem in the context of tag systems. However, the gap between the solvable classes and those that contain a universal tag system, is relatively large as compared to that for Turing machines. The smallest universal tag systems known are those that can be constructed through the Cocke-Minsky scheme. This scheme was provided to prove the recursive unsolvability of the class of tag systems with $v = 2$, improving the result from [23], but did not take into account the number of symbols μ. As a result, the universal tag systems that can be constructed through this scheme are still quite large: for any Turing machine with 2 symbols and m states, one needs a tag system with $v = 2$ and $\mu = 32m$. As

[5] I am indebted to Pascal Michel for pointing out this problem to me.

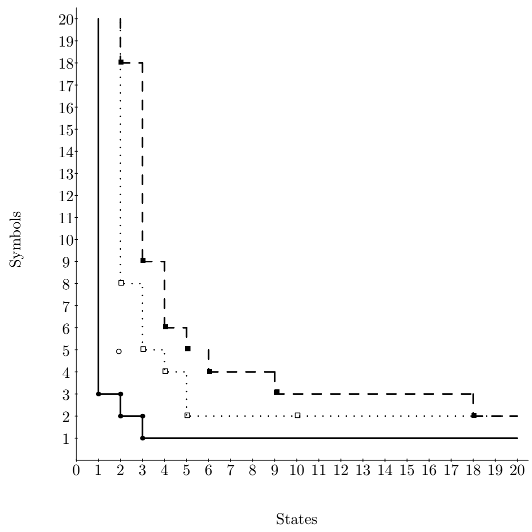

Figure 1. Limits of solvability and unsolvability in Turing machines. A full line denotes the solvability line, the dotted line the current $3n+1$-line, and the dashed line is the current unsolvability line.

a consequence, the tag system that can be constructed using this scheme in order to simulate the smallest 2-symbol universal Turing machine (which has 18 states) is in the class TS(576, 2).

Reducing the $3n + 1$ problem to a class in between the known recursively solvable and unsolvable classes, provides us with new information about such classes. It shows that a proof of the solvability of these classes depends on the $3n+1$-problem. This implies that proving these classes recursively solvable will be very hard. Such reductions were done for Turing machines by Baiocchi (mentioned in [18]), Margenstern [18] and Michel [21]. Michel also reduced several other Collatz-like problems to different Turing machines [22]. Margenstern calls the line formed in the state-symbol diagram by those machines to which the $3n+1$-problem can be reduced, the present $3n+1$-line, and conjectured that all points on the $3n+1$-line contain a machine with an undecidable halting problem or an undecidable reachability problem or an undecidable modified reachability problem (a conjecture that assumes of course nothing about the status of the $3n+1$-problem).

In Fig. 1 a summary is given of the known boundaries of solvability and unsolvability in Turing machines, including the $3n+1$-line. Fig. 2 gives an overview of the present situation of the boundaries of solvability and unsolvability in tag systems.

From Fig. 2 it is clear that the gap between the solvable class TS(2, 2) and the universal class TS(576, 2) is significantly larger as compared to that for Turing machines. The reduction of the Collatz problem to a small tag system in the class TS(3,2) – which has only one symbol more than the class TS(2, 2) – however shows that already on this low level one is confronted with problems related to an intricate problem of number theory. It will thus be very difficult to prove this class recursively solvable. As is furthermore clear from Figs. 1

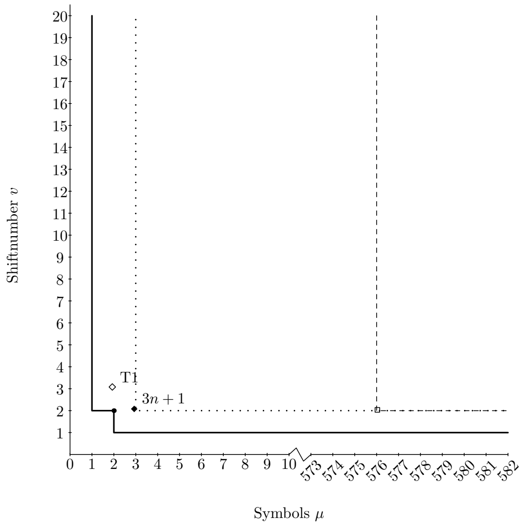

Figure 2. Limits of solvability and unsolvability in Turing machines. A full line denotes the solvability line, the dotted line the current $3n+1$-line, and the dashed line is the current unsolvability line. T1 is the tag system that was given as an example by Post.

and 2, the present $3n+1$-line in tag systems is lower than that for Turing machines. Indeed, whereas the class of tag systems TS(3, 2) contains T_C, the class of Turing machines TM(3, 2) is known to have a recursively solvable halting problem. This, together with the relatively small distance between the $3n+1$- and universality line in Turing machines, suggests that the present unsolvability line in tag systems might be seriously decreased. The intractability of the very simple tag system in the class TS(2, 3) as encountered by several researchers – including Post – only adds strength to this idea.
Of course, one should be very careful in drawing conclusions on the basis of a direct comparison between symbol-state complexity for Turing machines and symbol-shift complexity for tag systems. The small distance between the present universality- and $3n+1$-line in Turing machines together with the fact that the $3n+1$-line in tag systems is lower than that for Turing machines, does not necessarily imply that the distance between the universality- and $3n+1$-line in tag systems should be of a scale similar to that for Turing machines. [6]
Notwithstanding this problem, in the light of the present reduction, it seems quite improbable that the distance between the present $3n+1$- and unsolvability line in tag systems could not be seriously decreased. In fact, following Margenstern, we would like to propose the following conjecture:

Conjecture 1 *There exists at least one tag system with an unsolvable halting problem or an unsolvable reachability problem in every set of tag systems for which* $\mu > 2, v = 2$

[6] We are indebted to an anonymous referee for pointing out this problem to us.

As is the case for Margenstern's conjecture, this conjecture of course does not assume anything about the status of the $3n+1$-problem itself.
To conclude, the reduction of the $3n+1$-problem to a tag system in the class TS(3, 2) shows that proving the recursive solvability for tag systems in this class, i.e., to improve the result from [28] by increasing μ, will be very difficult. The reduction furthermore indicates that the present unsolvability line in tag systems might be considerably decreased.
It is suggested here that, given, on the one hand, the intensive use of tag systems in the research context of searching for small universal systems, and, on the other hand, their formal simplicity, they are interesting systems to be studied for themselves. There are several interesting open problems connected to tag systems, one of them being the search for smaller universal tag systems. Indeed, if the conjecture made here turns out to be true, one might perhaps find one of the simplest universal systems known. But, especially in connecting tag systems to number theory,[7] as was done in this paper, research on these systems seems promising.

References

[1] Claudio Baiocchi, *Three small universal Turing machines*, Machines, Computations and Universality MCU 2001 (M. Margenstern and Yu. Rogozhin, eds.), Lecture notes in Computer Science, vol. 2055, 2001, pp. 1–10.

[2] John Cocke and Marvin Minsky, *Universality of tag systems with p = 2*, 1963, Artificial Intelligence Project – RLE and MIT Computation Center, memo 52.

[3] John H. Conway, *Unpredictable iterations*, Proceedings of the 1972 Number Theory conference, 1972, pp. 49–52.

[4] ______, *Fractran- a simple universal computing language for arithmetic*, Open Problems in Communication and Computation (New York) (T.M. Coper and B. Gopinath, eds.), Springer Verlag, 1987, pp. 3–27.

[5] Matthew Cook, *Universality in elementary cellular automata*, Complex Systems **15** (2004), no. 1, 1–40.

[6] Martin Davis, *The undecidable. Basic papers on undecidable propositions, unsolvable problems and computable functions*, Raven Press, New York, 1965, Corrected republication (2004), Dover publications, New York.

[7] ______, *Why Gödel didn't have Church's thesis*, Information and Control **54** (1982), 3–24.

[8] ______, *Emil L. Post. His life and work.*, Solvability, Provability, Definability: The collected works of Emil L. Post [36], 1994, pp. xi–xviii.

[7] A relation already pointed out by Post [35].

[9] Jeremy Fox (ed.), *Mathematical theory of automata*, Microwave Research Institute Symposia Series, vol. XII, Brooklyn, NY, Polytechnic Press, 1963.

[10] Brian Hayes, *Theory and practice: Tag-you're it*, Computer Language (1986), 21–28.

[11] ———, *A question of numbers*, American Scientist **84** (1996), 10–14, Available at: http://www.americanscientist.org/amsci/issues/Comsci96/compsci96-01.html.

[12] F. Kaščák, *Small universal one-state linear operator algorithm*, Mathematical foundations of Comuter Science (L.M. Havel and V. Koubek, eds.), Lecture notes in Computer Science, vol. 629, 1992, pp. 327–335.

[13] M. Kudlek and Yurii Rogozhin, *New small universal circular Post machines*, Proceedings of the 13th International Symposium on Fundamentals of Computation Theory, Riga, Latvia, August 22-24, 2001, Lecture notes in computer science, vol. 2138, 2001, pp. 217–226.

[14] ———, *A universal Turing machine with 3 states and 9 symbols*, Proc. 5th International Conference on Developments in Language Theory (G. Rozenberg W. Kuich and A. Salomaa, eds.), Lectore Notes in Computer Science, vol. 2295, 2002, pp. 311–318.

[15] Jeffrey C. Lagarias, *The 3x + 1 problem and its generalizations*, American Mathematical Monthly **92** (1985), no. XX, 3–23, Available at: http://www.cecm.sfu.ca/organics/papers/lagarias/paper/html/paper.html.

[16] ———, *The 3x+1 problem and its generalisations*, Organic Mathematics. Proceedings Workshop Simon Fraser University, Burnaby (Providence) (J. Borwein et al., ed.), AMS, 1995, Available at http://www.cecm.sfu.ca/organics/papers/lagarias.

[17] ———, *The 3x + 1 problem: An annotated bibliography (1963–2000)*, 2006, Available at: http://arxiv.org/PS-cache/math/pdf/0608/0608208.pdf.

[18] Maurice Margenstern, *Frontier between decidability and undecidability: A survey*, Theoretical Computer Science **231** (2000), no. 2, 217–251.

[19] ———, *An Algorithm for Buiding Inrinsically Universal Automata in Hyperbolic Spaces*, Proceedings of the 2006 International Conference on Foundations of Computer Science (Hamid R. Arabnia Mark Burgin, eds.), Las Vegas, USA, 2006, 3–9.

[20] Sergei. J. Maslov, *On E. L. Post's 'Tag' problem. (russian)*, Trudy Matematicheskogo Instituta imeni V.A. Steklova (1964b), no. 72, 5–56, Translated in English in *American Mathemtical Society Translations, series 2*, vol. 97, nr. 2, 1970, 1 – 14.

[21] Pascal Michel, *Busy Beaver competition and Collatz-like problems*, Archive for Mathematical Logic **32** (1993), no. 5, 351–367.

[22] ———, *Small Turing machines and generalized Busy Beaver competition*, Theoretical Computer Science **326** (2004), no. 1–3, 45–56.

[23] Marvin Minsky, *Recursive unsolvability of Post's problem of tag and other topics in the theory of Turing machines*, Annals of Mathematics **74** (1961), 437–455.

[24] ———, *Universality of (p = 2) tag systems and a 4 symbol 7 state universal Turing machine*, 1961/62?, Artificial Intelligence Project – RLE and MIT Computation Center, memo 33.

[25] ———, *Size and structure of universal Turing machines using tag systems: a 4-symbol 7-state machine*, Proceedings Symposia in Pure Mathematics, American Mathematical Society **5** (1962), 229–238.

[26] ———, *Computation. Finite and infinite machines*, Series in Automatic Computation, Prentice Hall, Englewood Cliffs, New Jersey, 1967.

[27] Liesbeth De Mol, *Closing the circle: An analysis of Emil Post's early work*, The Bulletin of Symbolic Logic **12**, (2006), no. 2, 267–289.

[28] ———, *Study of Limits of Solvability in Tag Systems*, Machines, Computations and Universality, MCU 2007 (J. Durand-Lose and M. Margenstern, eds.), Lecture notes in Computer Science, vol. 4664, 2007, pp. 170–181.

[29] ———, *Solvability of the Halting and Reachability Problem for Tag systems with* $\mu = v = 2$, Preprint 343, Center for Logic and Philosophy of Science, Universiteit Gent, available at: http://logica.ugent.be/centrum/writings/pubs.php.

[30] ———, *Tracing Unsolvability. A historical, mathematical and philosophical analysis with a special focus on tag systems.*, Ph.D. thesis, Universiteit Gent, available at: http://logica.ugent.be/centrum/writings/doctoraten.php, 2007.

[31] Damien Woods and Thurlough Neary, *On the time complexity of 2-tag systems and small universal Turing machines*, FOCS. 47th Annual IEEE Symposium on Foundations of Computer Science, 2006, pp. 132–143.

[32] Thurlough Neary and Damien Woods, *Four small Universal Turing Machines*, Machines, Computations and Universality, MCU 2007 (J. Durand-Lose and M. Margenstern, eds.), Lecture notes in Computer Science, vol. 4664, 2007, pp. 242–254.

[33] Turlough Neary, *Small Polynomial time universal Turing machines*, Proceedings of MFCSIT 2006, Cork, Ireland, (T. Hurley, A. Seda et al., eds.) 2006, pp. 325–329.

[34] Emil Leon Post, *Formal reductions of the general combinatorial decision problem*, American Journal of Mathematics (1943), no. 65, 197–215.

[35] ———, *Absolutely unsolvable problems and relatively undecidable propositions - Account of an anticipation*, [6], 1965, pp. 340–433.

[36] ______, *Solvability, provability, definability: The collected works of Emil L. Post*, Birkhauser, Boston, 1994, edited by Martin Davis.

[37] Yurii Rogozhin, *Small universal Turing machines*, Theoretical Computer Science **168** (1996), 215–240.

[38] ______, *Seven universal Turing machines (in Russian)*, Mat. Issledovaniya **69** (1982), 76–90.

[39] Hao Wang, *Tag systems and Lag systems*, Mathematische Annalen **152** (1963), 65–74.

[40] Shigeru Watanabe, *Periodicity of Post's normal process of tag*, [9], 1963, pp. 83–99.