Date: 2025-12-30

# Turing Machines

**Abstract**

Developments are given here for the analysis technique for non-terminating Turing Machines (TM's) that I described earlier in [**?**] and [**?**]. The main new ideas are the introduction of IRR patterns i.e. constraints satisfied by large sets of IRRs (Irreducible Regular Rules) and the logical relationships between them as a result of the general method for deriving IRR's from others described in my earlier paper. These logical relationships will be referred to as IGR's (IRR Generating Rules). IGR's have been reduced to their minimal form in a way analogous to the way in which regular rules were reduced to IRR's by taking out symbol strings that played no essential role. In the case of IGR's these symbol strings (actually pairs) will be referred to as context pairs. A new version of my computer program extending the previous analysis is described and is freely available that generates these IGR's up to a given length of IRR's that they generate. The results show repetition of the left hand halves (Left IGR's or LIGR's) of IGR's associated with different right hand halves. Because the LIGR's can be derived independently of the right hand halves of IGR's, this should be done separately and can be done using the currently known IRR's as previously described in my earlier papers. The LIGR's can be used to calculate all the IRR's of a TM. A procedure for the generation of all the LIGR's for a TM has been suggested and is expressed here by a detailed analysis of a TM though not yet as computer code.

**Mathematics Subject Classification:** 68Q25

**Keywords:** Turing Machine (TM), Irreducible Regular Rule (IRR), IRR Generating Rule (IGR), LIGR (Left IGR).

# 1 Introduction

I plan this is the last version that includes a lot of material that probably will not be needed. It will be available as an old version but hopefully a much shortened version will take its place as the latest version under active editing.

I think there will be a long section that could be in an appendix that is not needed initially at least.

The diagram Figure **??** does not have a complete description of the inter-relationships between the loops indicated in parentheses. This is contained

in (**??**), (**??**) and (**??**) which were updated together with some text on page 59. Also Figure **??** does not have the result (**??**) which is equivalent to Table **??** which is also summarised by Figure **??** and shows how the TM can be 'trapped' in a steady movement to its left. The results (**??**), (**??**), (**??**) and (**??**) seem to be adequately describing the TM. (**??**) can only be deduced in this paper once attention is drawn to the vacuosity of LIGR's with LHS $3b^5\underline{d}$ which in turn arises out of the analysis of the completeness of the LIGR's (the very long proof). There could of course be other ways of getting this result but this seems to be the only systematic way to do it.

Once some basics are understood, probably the reader can jump to around page 59.

Table **??** is now hopefully complete and correct and the following summary **??**. Things to be corrected:

- section **??** needs some discussion because the two examples are so different.

Some general heuristics are given right at the end regarding using the IGR's to generate a description of the action of a TM in terms of ever expanding cycles (when possible). The TM (**??**) is an example of this while TM (**??**) is not.

Table **??** was reorganised (Table **??**) in terms of LIGR's and RIGR's(Right IGR's). This document is a work in progress. As such it is incomplete and still has errors and omissions. When brought to a state where I cannot easily find any improvements it will form my next paper on Turing Machine analysis.

Section 2 is a quite dense summary of the previous methods that lays the foundations of the developments to be described in this paper. Section 3 introduces IRR patterns (IRRP's) as sets of IRR's conforming to the pattern. They have some common symbols in the origin and the RHS of the IRR and allow for any LHS. Section 4 introduces IGR's in terms of IRRP's and illustrates the fact that IRR's of any length can be derived from sequences of IGR's by a sequence of substitutions. In Section 5 the detailed description of an IGR is given and proves the generation of IRR's from IGR's. A computer algorithm is described for generating them all up to a given length for a TM and its results are shown for an example TM. In Section 6 a necessary condition in the relation "can be followed by" for IGR's was found. Further results are found for the set of IGR's that can follow a sequence of IGR's following each other (i.e. substituted into each other) hand calculation of which suggests a method for generating all of them for a TM though this appears practically impossible for the example TM because of the large number of cases to be considered. In Section 7 left IGR's or LIGR's are introduced because in section 6 the LHS and RHS of an IGR can be developed independently. An algorithm is illustrated by example for finding all the LIGR's for a TM based on the above ideas and results.

A lot of material has been removed to 2017's Notes on Turing Machines. These notes are now mostly superseded, but there may be a little there that is of use.

Comments are welcome. Please send them to john.h.nixon1@gmail.com

# 2 Basic definitions and summary of the existing method (F) to generate the IRR's for a TM

A configuration set (CS) for a TM is a set of complete configurations (tape symbols with pointer position indicated, and the machine state of the TM) such that the CS is specified by giving a finite set of symbols in a set of contiguous pointer positions together with the machine state and such that the pointer position is where one of the given symbols is given or adjacent to one. In a CS all possible configurations that are consistent with the specified symbols and machine state are included. The notation is the specified symbol string with the pointer indicated by an underscore (it is just off the end of the symbol string) or an underline and the machine state on the left. For example with machine states 1,2,3, etc. and symbols lower case letters the following are CSs: `2ab_bca`, `1_aabbcac`. The length of the CS is the length of the symbol string which is finite.

A computation rule or rule is a pair of CS's linked by $\rightarrow$ indicating the forward direction of the computation. A reducible rule is one that has symbols that play no part in the computation i.e. any extra symbols added on the left or right of the strings at the left and right hand sides of a computation rule. From the definitions of regular rules (RR) and irreducible regular rules (IRR) in [**?**], any computation of the TM that ends with the pointer just off the end (i.e. adjacent to a symbol at the end) of the string of symbols specified at the start can be represented by RR's chained together as a sequence of CS's starting with one of length 1, where for each step in the chain a new symbol is read at a position where no symbol has yet been read at the pointer, thus the length of the string of symbols increases by 1 for each RR unless a stationary cycle occurs that ends such a sequence. All such CS's are by definition reachable. All single TM steps are RR's. If the RR is of type LR or RL as designated earlier (now the position of the pointer in the origin (see below) is included so these are now RLR and LRL respectively) the pointer swaps ends at that step of the chain and these RR's are also irreducible RR's (IRR's) because if the pointer swaps ends there are no redundant symbols i.e. the rule is irreducible. There are also IRR's that that don't swap ends. If a CS called "origin" is included with the LHS and RHS of the IRR it can be written in the triplet form as `origin` $\rightarrow$ `LHS` $\rightarrow$ `RHS` for which the abbreviated form `origin` $\rightarrow\!\!\rightarrow$ `RHS` will

be used if the LHS is not specified hence the changed designation of the type of an IRR. An origin (there could be many for the same LHS) of an IRR is a CS obtained by running the TM backwards starting from the LHS to a point such that the pointer position is at the opposite end of the string from where it is in the LHS.

If an RR is of type LRR or RLL it is related to an irreducible form (a possibly shorter IRR which only involves the symbols passed by the pointer during its execution) as follows. Suppose the RR is represented by $\mathtt{m} \to \mathtt{n} \to \mathtt{o}$ where $m < n < o$ and $n + 1 = o$ where italics represent the corresponding pointer positions for the CS's in typewriter font. Then the RR has type LRR because the start and end points of the i.e. the LHS and RHS have the pointer at the right hand end of the string. The rule $\mathtt{n} \to \mathtt{o}$ can be represented as $\mathtt{n}' \to \mathtt{p}' \to \mathtt{o}'$ without any redundant symbols where $m \leq p \leq n < o$ and the primes indicate shortening of the strings by deleting the symbols below position $p$ i.e. $p$ is the leftmost pointer position in the computation from $\mathtt{n}$ to $\mathtt{o}$. $n = p$ holds if and only if $\mathtt{n}' = \mathtt{p}'$ and $\mathtt{n}' \to \mathtt{o}'$ is a single TM step. If $n \neq p$ the rule $\mathtt{n}' \to \mathtt{p}'$ shows that $\mathtt{p}'$ is reachable therefore $\mathtt{n}' \to \mathtt{p}' \to \mathtt{o}'$ represents an IRR of type RLR, and of course the mirror image result applies to IRR's of type RLL.

In general let $\mathtt{X}$ be a member of IRR($n$) i.e. the set of all IRR's with CS's of length $n$. Then $\mathtt{X}$ can be represented as $\mathtt{A} \to \mathtt{B} \to \mathtt{C}$ where the pointer swaps ends between $\mathtt{A}$ and $\mathtt{B}$ (thus this is either $\mathtt{1} \to \mathtt{n}$ or $\mathtt{n} \to \mathtt{1}$ and is referred to as condition 1) to establish the reachability of $\mathtt{B}$ necessary for $\mathtt{X}$ to be an IRR. There may be more than one such CS $\mathtt{A}$ for a single $\mathtt{B}$ and the set of all such $\mathtt{A}$ will be denoted by $\mathtt{O_1(B)}$ (the same as $\mathtt{S(B)}$ in [**?**]), the 1 referring to the backward searching algorithm that terminates in condition 1 (see [**?**] section 2.2). Likewise if it terminates in condition 2 i.e. the pointer comes back to where it was at the start of the backward search, the set of such endpoints will be denoted as $\mathtt{O_2(B)}$, but these do not confer reachability and will not be referred to as origins. If the pointer is at the right in $\mathtt{A}$ and at the left in $\mathtt{B}$ then at $\mathtt{C}$ it can be at the left or right so that $\mathtt{X}$ must be represented as either of the triplet forms $\mathtt{n} \to \mathtt{1} \to \mathtt{0}$ or $\mathtt{n} \to \mathtt{1} \to \mathtt{n} + \mathtt{1}$ and having types RLL and RLR respectively. Likewise if the pointer is at the left in $\mathtt{A}$ (the mirror image forms), $\mathtt{X}$ must be represented by either of $\mathtt{1} \to \mathtt{n} \to \mathtt{n} + \mathtt{1}$ or $\mathtt{1} \to \mathtt{n} \to \mathtt{0}$, having types LRR and LRL respectively.

If $\mathtt{B}$ is reachable but forward computation from it leads to a CS that has arisen before in this computation, this is a stationary cycle and the type of the IRR is then LC or RC. If the reverse computation path from $\mathtt{B}$ leads to a stationary cycle, then this cycle must include $\mathtt{B}$ to avoid a branch point in the forward computation that would not then be unique. Thus likewise the IRR is of type LC or RC.

From the definition of RR in the first paragraph of this section, if in the

backward search from the LHS of an IRR, the pointer again reaches the same position it had in the LHS (condition 2), however much further back the backward search were to continue, it would not be possible from this alone to show that this LHS is indeed the LHS of an IRR. This is because if the computation is again run forward, this LHS has the pointer at the same point as a previous CS and is therefore not shown to be one of the list of CS's playing the special role in the above definition, though it could possibly be shown to be one as a result of another backward search path. This is the justification of the terminating condition 2 in the backward search algorithm. See [**?**] page 30.

This proves that

**Lemma 2.1.** *The triplets* $\mathtt{1} \to \mathtt{n} \to \mathtt{0}$, $\mathtt{1} \to \mathtt{n} \to \mathtt{n}+\mathtt{1}$, *and* $\mathtt{n} \to \mathtt{1} \to \mathtt{n}+\mathtt{1}$, *and* $\mathtt{n} \to \mathtt{1} \to \mathtt{0}$ *representing TM computations each form an IRR* (*type LRL, LRR, RLR or RLL respectively*) *if and only if the origin indicated* (*the first member*) *is the first CS arrived at with the pointer in that position after tracing the computation back from the LHS* (*the middle member*) *and the pointer does not occur again in the position it had in the LHS in the reverse computation path from the LHS i.e there is no other CS* $\mathtt{1}$ *or* $\mathtt{n}$ *between the* $\mathtt{1}$ *and the* $\mathtt{n}$ *in the triplet forms above. Furthermore any IRR of length n of one of the types RLL, RLR, LRL, LRR has one of these forms. Note that because the symbol strings here are of length n having symbols at positions k such that* $1 \leq k \leq n$, *the CS's* $\mathtt{0}$ *and* $\mathtt{n}+\mathtt{1}$ *are necessarily the first CS's reached with the pointer in positions 0 and n+1 respectively.*

Generating all the IRR's based on Theorem 9.1 of [**?**] starts with all single TM steps in the above notation $\mathtt{1} \to \mathtt{0}$ (i.e. $\underline{\mathtt{x}} \to \_\mathtt{x}$) or $\mathtt{1} \to \mathtt{2}$ (i.e. $\underline{\mathtt{x}} \to \mathtt{x}\_$) where $\mathtt{x}$'s represents an arbitrary symbol that could be different for each use. Every possible single symbol (called $\alpha$) is added at the pointer position in each RHS, and in each case the computation is taken as far as possible to get the new RHS unless a stationary cycle occurs. The resulting rule $\mathtt{LHS} \to \mathtt{RHS}$ is an IRR if it irreducible i.e. cannot be expressed with shorter strings of symbols. In the first case, adding $\alpha$ on the left and continuing the computation as far as possible gives results either of the form (i) $\mathtt{2} \to \mathtt{1} \to \mathtt{3}$ or (ii) $\mathtt{2} \to \mathtt{1} \to \mathtt{0}$ i.e. $\alpha\underline{\mathtt{x}} \to \underline{\alpha}\mathtt{x} \to \mathtt{xx}\_$ or $\_\mathtt{xx}$ respectively unless a stationary cycle is obtained. The results in case (i) are IRR's by Lemma **??** because there can be no other CS's between the $\mathtt{2}$ and the $\mathtt{1}$ which is a single TM step. The results from (ii) are IRR's if and only if the first move beyond the $\mathtt{1}$ is to $\mathtt{2}$ i.e. the computation has the form $\mathtt{2} \to \mathtt{1} \to \mathtt{2} \to \mathtt{0}$ because this ensures that the rule $\mathtt{1} \to \mathtt{0}$ contained in (ii) of length 2 is irreducible. Likewise for the mirror image case starting with a rule of the form $\mathtt{1} \to \mathtt{2}$ adding the $\alpha$ on the right and continuing gives results of the form $\mathtt{1} \to \mathtt{2} \to \mathtt{0}$ ( $\in$ IRR($\mathtt{2}$)) or of the form $\mathtt{1} \to \mathtt{2} \to \mathtt{3}$ ( $\in$ IRR($\mathtt{2}$) if and only if the first move from the $\mathtt{2}$ is to $\mathtt{1}$).

Consider extending this to the general case of generating all the IRR $\mathtt{Y}$ of length $n+1$ based on the single IRR $\mathtt{X}$ of length $n \geq 2$ and having the form

$\mathtt{n} \to \mathtt{1} \to \mathtt{n+1}$, which can be also be written as $\mathtt{A} \to \mathtt{B} \to \mathtt{C}$ for some CS's A,B, and C. First the computation $\mathtt{A}\alpha \to \mathtt{B}\alpha \to \mathtt{C}\alpha$ holds where $\alpha$ is any symbol the TM uses. Clearly by Theorems 5.4 and 9.1 of [**?**] every such IRR Y can be obtained starting from the LHS $\mathtt{B}\alpha$ if an appropriate $\alpha$ can be found. The symbol $\alpha$ must be chosen so that $\mathtt{B}\alpha$ is reachable i.e. $\mathtt{O_1}(\mathtt{B}\alpha) \neq \emptyset$. These are all the terminal CS's of length $n+1$ from the backward searching algorithm starting from $\mathtt{B}\alpha$ and ending in condition (1). Each of these branches has a point where the pointer first reaches n and this CS is $\mathtt{A}\alpha$ because the $\alpha$ has yet played no part, so $\mathtt{O_1}(\mathtt{B}\alpha) = \mathtt{O_1}(\mathtt{A}\alpha)$, thus the backward search algorithm is applied to $\mathtt{A}\alpha$, and identifying all possible values of $\alpha$ i.e. the values of $\alpha$ for which $\mathtt{O_1}(\mathtt{A}\alpha) \neq \emptyset$ by generating all its origins for each such $\alpha$. Also the forward computation from $\mathtt{C}\alpha$ is continued as far as possible to generate the RHS of Y and hence what its type is (LRR, LRL, RLR, RLL, LRC, or RLC) the last two cases coming from a stationary cycle in the forward computation from $\mathtt{C}\alpha$.

If the pointer is at the left in A and C and the right in B (the mirror image case) the added arbitrary symbol $\alpha$ will be on the left. This procedure for generating the all the IRR Y of length $n+1$ like this from an IRR X of length $n$, including the mirror image case where the triplet form of X is $\mathtt{1} \to \mathtt{n} \to \mathtt{0}$, will denoted by the function F. F applied to an IRR of type LC or RC is the empty set. This proves that

**Theorem 2.2.** *Every member of IRR$(n+1)$ can be obtained using* F *from some* $\mathtt{X} \in$ *IRR$(n)$ of type RLR or LRL for $n \geq 2$. Also, because forward computation is unique e.g. the RHS of an IRR is uniquely determined by is LHS, but the origins may be more than one, the sets of IRR obtained like this for different* X (*different* B) *must be disjoint i.e.* $\mathtt{F}^{-1}$ *applied to a member of IRR$(n+1)$ is a unique member of IRR$(n)$. The first part can be written symbolically as*

$$IRR(n+1) = \bigcup_{\mathtt{X} \in \mathtt{IRR}(n)} \{\mathtt{F(X)}\} \tag{1}$$

The following is the general outline showing an IRR triplet of length $n$ of type RLR (type RLR with origin having the pointer at the right) and the possible types of result (except the cases where a stationary cycle occurs) of this argument for a given symbol $\alpha$ (added on the right) that could include a new IRR triplet of length $n+1$.

$$\left.\begin{array}{l}\text{Cannot be used to}\\ \text{prove reachability of } \mathtt{1}\\ \text{Proves reachability of } \mathtt{1}\end{array}\right\} \begin{array}{rr} \mathtt{1} & \to \\ \mathtt{n+1} & \to \end{array} \Bigg\} \mathtt{n} \to \mathtt{1} \to \mathtt{n+1} \left\{ \begin{array}{rrl} \to & \mathtt{0} & \text{type RLL} \\ \to & \mathtt{n+2} & \text{type RLR} \end{array} \right. \tag{2}$$

The 3 central CS's refer to a member of IRR$(n)$, and the leftmost, central, and rightmost CS's refer to the corresponding member of IRR$(n+1)$ if reachability

of the CS 1 in the centre of (??) is found. The corresponding mirror image result for the LRL case is as follows:

$$\left.\begin{array}{l}\text{Proves reachability of } \mathtt{n+1} \quad \mathtt{1} \quad \rightarrow \\ \text{Cannot be used to prove} \\ \text{reachability of } \mathtt{n+1} \qquad \mathtt{n+1} \rightarrow \end{array}\right\} \mathtt{2} \rightarrow \mathtt{n{+}1} \rightarrow \mathtt{1} \left\{\begin{array}{lll} \rightarrow & \mathtt{0} & \text{type LRL} \\ \rightarrow & \mathtt{n+2} & \text{type LRR} \end{array}\right. \tag{3}$$

In this case note that because the $\alpha$ is added on the left, all the pointer positions in the IRR of length $n$ have been increased by 1 when they appear in the embedded IRR of length $n$, so originally they would have been $\mathtt{1} \rightarrow \mathtt{n} \rightarrow \mathtt{0}$. The above procedure allows the generation of all the IRR of a TM up to any given length and has been implemented [?].

Because the middle element of an IRR X ($\mathtt{A} \rightarrow \mathtt{B} \rightarrow \mathtt{C}$) only has a single symbol added when doing F, its inverse can be described as follows for the case that the IRR has type LRL. Take the middle element and remove its leftmost symbol giving $\mathtt{B}^*$, then trace the computation both back to find the first CS with the pointer at position 2 giving $\mathtt{A}^*$, and forward to get the first CS with the pointer at position 1 giving $\mathtt{C}^*$. This ensures that $\mathtt{A}^* \rightarrow \mathtt{B}^* \rightarrow \mathtt{C}^*$ satisfies Lemma ??. Then $\mathtt{F}^{-1}(\mathtt{X}) = \mathtt{A}^* \rightarrow \mathtt{B}^* \rightarrow \mathtt{C}^*$ and similarly for IRR's of type RLR. If only the origin and RHS of X is given as when an IRR pattern is provided, first an LHS must be found such that Lemma ?? holds.

# 3 Introducing IRR patterns (IRRP's) and IRR generating rules (IGR's)

The derivation of IRR's from other ones (length $n$) following the procedure F described above was found to often take the same form independent of $n$ provided $n$ is large enough. Then the obvious step is to describe these general results termed IRR generating rules (IGR's) so that they can be easily applied in any given case. These results have an LHS, the symbol $\alpha$ used in the derivation F, and an RHS that can match IRR's, and the existence of a member of IRR($n$) matching the LHS implies the existence of a corresponding member of IRR($n$+1) matching the RHS (or each of the parts of the RHS where there are more than one of them). The matching implies fixing the arbitrary strings $\mathtt{T_1}$ and $\mathtt{T_2}$. The general notation for an IGR is $\mathtt{LHS} \stackrel{\alpha}{\Rightarrow}$ {set of RHS's}. Each of these RHS's and the LHS take the form of a generalised IRR (an IRRP or IRR pattern) in which two arbitrary strings appear, $\mathtt{T_1}$ in the origin and $\mathtt{T_2}$ in the RHS (of the IRRP), and the LHS (middle member of the IRRP) is omitted so that any LHS (of the IRRP) is matched.

The analysis techniques were initially applied to the following TM (??) which was generated randomly with 5 states and 5 symbols. This TM, being much larger than any that I have analysed before, has proved to be a much

more challenging case.

$$\begin{array}{lllll} 1\underline{a} \to 2\_d & 2\underline{a} \to 1c\_ & 3\underline{a} \to 4c\_ & 4\underline{a} \to 3\_b & 5\underline{a} \to 2\_e \\ 1\underline{b} \to 4\_d & 2\underline{b} \to 4\_c & 3\underline{b} \to 4\_c & 4\underline{b} \to 4b\_ & 5\underline{b} \to 3\_e \\ 1\underline{c} \to 3\_a & 2\underline{c} \to 1d\_ & 3\underline{c} \to 2\_a & 4\underline{c} \to 3c\_ & 5\underline{c} \to 3a\_ \\ 1\underline{d} \to 2b\_ & 2\underline{d} \to 1a\_ & 3\underline{d} \to 5\_c & 4\underline{d} \to 5\_c & 5\underline{d} \to 4\_a \\ 1\underline{e} \to 2b\_ & 2\underline{e} \to 3\_c & 3\underline{e} \to 3b\_ & 4\underline{e} \to 5a\_ & 5\underline{e} \to 3a\_ \end{array} \tag{4}$$

For example it is known that at least one IRR for this TM matches

$$1\underline{d}aT_1 \to\to 4\_caT_2. \tag{5}$$

Using backward TM steps from (**??**) gives

$$\begin{array}{llll} \text{deriving the origin} & \text{old RHS} & \text{RHS} & \alpha \\ \alpha\mathtt{A} & \alpha\mathtt{C} & & \\ 1\alpha\underline{d}aT_1 \left\{ \begin{array}{l} \overset{\alpha=a}{\leftarrow} 2\underline{d}daT_1 \\ \overset{\alpha=c}{\leftarrow} 2\underline{a}daT_1 \\ \overset{\alpha=d}{\leftarrow} 2\underline{c}daT_1 \end{array} \right. & \begin{array}{l} 4\underline{a}caT_2 \\ 4\underline{c}caT_2 \\ 4\underline{d}caT_2 \end{array} & \begin{array}{l} 3\_bcaT_2 \\ 1abc\underline{T_2} \\ 5\_ccaT_2 \end{array} & \begin{array}{l} a \\ c \\ d \end{array} \end{array} \tag{6}$$

of which the result for $\alpha = \mathtt{c}$ has an RHS given where the pointer is at the first symbol of $T_2$. The results of F are written as follows

$$\begin{array}{l} 2\underline{d}daT_1 \to\to 3\_bcaT_2 \\ 2\underline{a}daT_1 \to\to 1abc\underline{T_2} \\ 2\underline{c}daT_1 \to\to 5\_ccaT_2 \end{array} \tag{7}$$

for $\alpha = \mathtt{a}, \mathtt{c}, \mathtt{d}$ respectively, and the complete IGR can be written as

$$1\underline{d}aT_1 \to\to 4\_caT_2 \left\{ \begin{array}{l} \overset{a}{\Rightarrow} 2\underline{d}daT_1 \to\to 3\_bcaT_2 \\ \overset{c}{\Rightarrow} 2\underline{a}daT_1 \to\to 1abc\underline{T_2} \\ \overset{d}{\Rightarrow} 2\underline{c}daT_1 \to\to 5\_ccaT_2 \end{array} \right. \tag{8}$$

Note that in this argument, adding the arbitrary symbol $\alpha$ on the left (because the pointer in the origin is on the left) maintains the pointer being on the right hand end of the string of symbols in the LHS (not shown), and this property is implicit in an IRRP with the pointer at the left of the origin CS. The result of this argument is that if an IRR of length $n$ conforms to (**??**), then there are 3 more IRR's of length $n+1$ corresponding to (**??**) for $\alpha \in \{\mathtt{a}, \mathtt{c}, \mathtt{d}\}$ respectively. The second of these results in (**??**) has the pointer in the RHS on the right, so this IRR has type LRR and cannot be used to derive other IRR's. These are examples of rules that generate IRR's of length $n+1$ from other IRR's of length $n$. An IGR is defined as a logical implication having an IRRP on the left and sets of IRRP's on the right, one set for each value of $\alpha$ and the

logical deduction follows the general procedure outlined in Theorem **??**. Thus the strings with given symbols on the right of the implications are one symbol longer than those on the left.

To illustrate how an IRR can be derived from a member of IRR(2) and a sequence of IGR's, consider the following IRR of length 6 that was chosen from the computer program output and represented in Origin → LHS → RHS format as follows:

$$\texttt{3}\underline{\texttt{a}}\texttt{ecccb} \rightarrow \texttt{1cadbd}\underline{\texttt{b}} \rightarrow \texttt{2dbdbdb\_}. \tag{9}$$

The derivation of the first rule of (**??**) in single TM steps is

$$\begin{array}{l} \texttt{3}\underline{\texttt{a}}\texttt{ecccb} \\ \texttt{4c}\underline{\texttt{e}}\texttt{ccccb} \\ \texttt{5ca}\underline{\texttt{c}}\texttt{ccb} \\ \texttt{3caa}\underline{\texttt{c}}\texttt{cb} \\ \texttt{2ca}\underline{\texttt{a}}\texttt{acb} \\ \texttt{1cac}\underline{\texttt{a}}\texttt{cb} \\ \texttt{2ca}\underline{\texttt{c}}\texttt{dcb} \\ \texttt{1cad}\underline{\texttt{d}}\texttt{cb} \\ \texttt{2cadb}\underline{\texttt{c}}\texttt{b} \\ \texttt{1cadbd}\underline{\texttt{b}} \end{array}. \tag{10}$$

Each time the pointer moves to where it has not been before while going backwards from the LHS, the derivation (**??**) generates IRR's as follows, followed by (**??**) in triplet and the abbreviated notation:

$$\begin{array}{ll} \texttt{2}\underline{\texttt{c}}\texttt{b} \rightarrow \texttt{1d}\underline{\texttt{b}} \rightarrow \texttt{5\_cd} \in \texttt{IRR}(2) & \texttt{2}\underline{\texttt{c}}\texttt{b} \rightarrow\!\!\rightarrow \texttt{5\_cd} \\ \texttt{1}\underline{\texttt{d}}\texttt{cb} \rightarrow \texttt{1bd}\underline{\texttt{b}} \rightarrow \texttt{3\_ecd} \in \texttt{IRR}(3) & \texttt{1}\underline{\texttt{d}}\texttt{cb} \rightarrow\!\!\rightarrow \texttt{3\_ecd} \\ \texttt{2}\underline{\texttt{c}}\texttt{dcb} \rightarrow \texttt{1dbd}\underline{\texttt{b}} \rightarrow \texttt{5\_cecd} \in \texttt{IRR}(4) & \texttt{2}\underline{\texttt{c}}\texttt{dcb} \rightarrow\!\!\rightarrow \texttt{5\_cecd} \\ \texttt{4}\underline{\texttt{e}}\texttt{cccb} \rightarrow \texttt{1adbd}\underline{\texttt{b}} \rightarrow \texttt{2\_ececd} \in \texttt{IRR}(5) & \texttt{4}\underline{\texttt{e}}\texttt{cccb} \rightarrow\!\!\rightarrow \texttt{2\_ececd} \end{array}. \tag{11}$$

This splitting up of the derivation of (**??**) results from the repeated application of `F` to (**??**).1. The abbreviated forms in (**??**) can be obtained by applying in order the following results to the first of these IRR's $\texttt{2}\underline{\texttt{c}}\texttt{b} \rightarrow\!\!\rightarrow \texttt{5\_cd}$.

$$
\begin{array}{l}
2\underline{\mathtt{T}}_1 \rightarrow\rightarrow 5\_\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} \left.\begin{array}{l} 1\underline{\mathtt{d}}\mathtt{T}_1 \\ 1\underline{\mathtt{e}}\mathtt{T}_1 \end{array}\right\} \rightarrow\rightarrow 3\_\mathtt{eT}_2 \\
1\underline{\mathtt{T}}_1 \rightarrow\rightarrow 3\_\mathtt{T}_2 \left\{\begin{array}{l} \overset{\mathtt{a}}{\Rightarrow} 2\underline{\mathtt{d}}\mathtt{T}_1 \rightarrow\rightarrow 4\mathtt{c}\underline{\mathtt{T}}_2 \\ \overset{\mathtt{c}}{\Rightarrow} 2\underline{\mathtt{a}}\mathtt{T}_1 \rightarrow\rightarrow 2\_\mathtt{aT}_2 \\ \overset{\mathtt{d}}{\Rightarrow} 2\underline{\mathtt{c}}\mathtt{T}_1 \rightarrow\rightarrow 5\_\mathtt{cT}_2 \end{array}\right. \\
2\underline{\mathtt{c}}\mathtt{dT}_1 \rightarrow\rightarrow 5\_\mathtt{T}_2 \overset{\mathtt{a}}{\Rightarrow} \left.\begin{array}{l} 4\underline{\mathtt{e}}\mathtt{ccT}_1 \\ 4\underline{\mathtt{e}}\mathtt{ecT}_1 \\ 5\underline{\mathtt{c}}\mathtt{adT}_1 \\ 5\underline{\mathtt{e}}\mathtt{adT}_1 \end{array}\right\} \rightarrow\rightarrow 2\_\mathtt{eT}_2 \\
4\underline{\mathtt{T}}_1 \rightarrow\rightarrow 2\_\mathtt{eT}_2 \overset{\mathtt{c}}{\Rightarrow} \left.\begin{array}{l} 3\underline{\mathtt{a}}\mathtt{T}_1 \\ 4\underline{\mathtt{b}}\mathtt{T}_1 \end{array}\right\} \rightarrow\rightarrow 2\mathtt{db}\underline{\mathtt{T}}_2
\end{array}
\quad . \tag{12}
$$

For the initial steps in the derivation of (**??**), the following sub-cases of successive members of (**??**) need to be applied in this order: 1, 3, 1, 1.

Equation (**??**) contains examples of IGR's which allow one IRR to be derived from another by substituting for the $\mathtt{T}_1$ and $\mathtt{T}_2$ as the example shows, and express the application of the function F in Theorem **??** in a simpler form. The IGR's have two lengths, one associated with $\mathtt{T}_1$ and one associated with $\mathtt{T}_2$ and these are defined as the lengths of the corresponding strings on the RHS of the IGR thus for example the lengths of the IGR's in (**??**) will be denoted by $(1,1),(1,1),(3,1),(1,2)$ respectively. The equations (**??**) are in the shortest forms possible as can be verified from their derivations.

The symbols above the implication signs are the symbol added next to the pointer in the origin ($\alpha$) in the derivation of the IRR's from other ones as described in Section **??**, and are the first 4 symbols of the LHS of IRR (**??**) taken in reverse order. The results on the right in (**??**) are all the results that can be derived from their LHS for that value of $\alpha$ and that length, though the third example is quite complicated and has other values of $\alpha$ ie. b and c with different lengths of results.

The IRRP on the RHS of the last member of (**??**) is of type LRR and can be seen to not generate a new IRR directly. Applying the last member of (**??**) to the last IRR of (**??**) gives the initial result

$$3\underline{\mathtt{a}}\mathtt{ecccb} \rightarrow 1\mathtt{cadbd}\underline{\mathtt{b}} \rightarrow 2\mathtt{db}\underline{\mathtt{c}}\mathtt{ecd} \tag{13}$$

which is not an IRR. Taking this computation as far a possible has to be an IRR (in this case having non-extendable type LRR) which is

$$3\underline{\mathtt{a}}\mathtt{ecccb} \rightarrow\rightarrow 2\mathtt{dbdbdb\_} \tag{14}$$

and has $\alpha = \mathtt{c}$ and is in agreement with (**??**). The derivations of (**??**) and (**??**) illustrate the general procedure for deriving any IRR by repeated applications of F i.e. applying a sequence of IGR's starting from a member of IRR(2).

This example suggests that if all the IGR's needed to generate the IRR($n+1$) from IRR($n$) were obtained, these could have lengths much less than $n+1$ and be fewer in number than the IRR($n+1$), and this might give a more compact way to represent the action of the TM. This will be followed up later, but many details need to be given first.

Every IGR represents the process of deriving a member of IRR($n+1$) from a member of IRR($n$). Therefore every such IGR can be obtained from another IGR (representing the process for deriving the member of IRR($n$) from a member of IRR($n-1$)) by an appropriate specialisation by adding the context symbols, applying F, then removing any redundant symbols as before. But the number of such context symbols needed seems to be unlimited.

# 4 General definitions of F and IGR's

The general form of the derivation of an IRR from an existing one (F) can be expressed in detail as follows. Start with the IRR pattern (IRRP) of type LRL

$$\mathtt{t}_1\underline{\mathtt{y}_1}\ldots\mathtt{y}_n\mathtt{T}_1 \rightarrow\!\!\rightarrow \mathtt{t}_2\_\mathtt{z}_1\ldots\mathtt{z}_n\mathtt{T}_2 \tag{15}$$

in which $\mathtt{T}_1$ and $\mathtt{T}_2$ have been omitted for brevity in much of this section. Here $n \geq 2$ and the t's are machine states and y's and z's are symbols.

Then proceed with F i.e. add the symbol $\alpha$ to both sides where the pointer is in the RHS then the backward search gives the following types of results (excluding the stationary cycles) which can be classified according to the rightmost position $j_1$ of the pointer relative to the symbol $\mathtt{y}_1$

$$\mathtt{t}_1\alpha\underline{\mathtt{y}_1}\ldots\mathtt{y}_n \leftarrow \begin{cases} \mathtt{t}'_1\underline{\alpha'}\mathtt{y}_1\ldots\mathtt{y}_n \text{ for } j_1 = 0 \\ \mathtt{t}'_1\underline{\alpha'}\mathtt{y}'_1\ldots\mathtt{y}'_{j_1+1}\mathtt{y}_{j_1+2}\ldots\mathtt{y}_n \text{ for } 1 \leq j_1 \leq n-2 \\ \mathtt{t}'_1\alpha\mathtt{y}'_1\ldots\mathtt{y}'_{n-1}\underline{\mathtt{y}'_n} \text{ for } j_1 = n-1 \end{cases} \tag{16}$$

where the primes indicate a possible change in the symbol or state by the TM. For the case $n = 1$, $j_1$ must be 0. Note that the form $\mathtt{t}'_1\underline{\alpha'}\mathtt{y}'_1\mathtt{y}_2\ldots\mathtt{y}_n$ cannot arise because a single backward step to the right followed by two backward steps to the left could possibly alter $\mathtt{y}_1$ and $\mathtt{y}_2$ whereas a single backward step to the left has $j_1 = 0$ as above.

The point of the classification is to enumerate all the different types of case that can arise after all the symbols that cannot be altered because the pointer does not reach there in the derivation, are abstracted out. They are not mentioned explicitly and they form part of an arbitrary string (in this case $\mathtt{T}_1$). The last reverse computation step in the last case giving $j_1 = n-1$ cannot not lead to a new IRR because this path and the CS reached does not imply the reachability of the LHS and so does not generate an IRR. If the LHS is

reachable it must be because there is another origin with $j_1 < n-1$. Therefore this case must be omitted for the purpose of generating IGR's, so $j_1$ can be restricted to the range $0 \leq j_1 \leq n-2$.

Similarly, for the computation of the new RHS, the results can be classified (again excluding stationary cycles) by the rightmost position $j_2$ of the pointer. So that this parameter also starts at 0, the pointer starts at position 0 at $\alpha$ and ends at position -1 if it goes left and ends at position $n+1$ if it goes right giving the possibilities

$$\mathtt{t}_2\underline{\alpha}\mathtt{z}_1\ldots\mathtt{z}_n \rightarrow \begin{cases} \mathtt{t}'_2\_\alpha'\mathtt{z}'_1\ldots\mathtt{z}'_{j_2}\mathtt{z}_{j_2+1}\ldots\mathtt{z}_n \text{ where } 0 \leq j_2 \leq n \text{ or} \\ \mathtt{t}'_2\alpha'\mathtt{z}'_1\ldots\mathtt{z}'_{n-} \text{ if } j_2 = n+1 \end{cases}. \tag{17}$$

This works whenever $n \geq 1$.

If a stationary cycle occurred in (**??**) it would be noted, but it would have no effect on the general form of the possible reverse search results. Because $\mathtt{t}_1\alpha\underline{\mathtt{y}_1}\ldots\mathtt{y}_n$ is in the closed circuit (to avoid a branch point in the forward computation implying it is not unique) the derived IRR would have type RC i.e. a stationary cycle occurs in the result of (**??**).

The minimum number of symbols needed for the representation of (**??**) is easily seen to be

$$r_1 = \begin{cases} 1 & \text{for } j_1 = 0 \\ j_1 + 2 & \text{otherwise} \end{cases} \tag{18}$$

provided $0 \leq j_1 \leq n-2$. Similarly, the minimum number of symbols needed for the representation of the result of (**??**) is

$$r_2 = \min(j_2 + 1, n+1). \tag{19}$$

The length of an IGR consists of the pair $(r_1, r_2)$.

From (**??**) and (**??**), because F only applies to an IRR of extendable type, i.e. type LRL in this case, and because RCS's resulting from the backward search going to the opposite end of the string from $\alpha$ are excluded from the LHS's of IGR's, and excluding the cycling cases described above, the remaining four combinations can be summarised as

$$\mathtt{t}_1\underline{\mathtt{y}_1}\ldots\mathtt{y}_n\mathtt{T}_1 \rightarrow\!\!\rightarrow \mathtt{t}_2\_\mathtt{z}_1\ldots\mathtt{z}_n\mathtt{T}_2 \stackrel{\alpha}{\Rightarrow}$$

$$\left.\begin{cases} \mathtt{t}'_1\underline{\alpha'}\mathtt{T}_1 & j_1 = 0 \\ \mathtt{t}'_1\underline{\alpha'}\mathtt{y}'_1\ldots\mathtt{y}'_{j_1+1}\mathtt{T}_1 & 1 \leq j_1 \leq n-2 \end{cases}\right\} \rightarrow\!\!\rightarrow \left\{\begin{cases} \mathtt{t}'_2\_\alpha'\mathtt{z}'_1\ldots\mathtt{z}'_{j_2}\mathtt{T}_2 & 0 \leq j_2 \leq n \\ \mathtt{t}'_2\alpha'\mathtt{z}'_1\ldots\mathtt{z}'_n\underline{\mathtt{T}_2} & j_2 = n+1 \end{cases}\right\}. \tag{20}$$

In this statement the top and bottom parts on the left of $\rightarrow\!\!\rightarrow$ can be combined independently with the top and bottom parts on the right of $\rightarrow\!\!\rightarrow$ i.e. there are four combinations possible. Of these the distinctions on the left of $\rightarrow\!\!\rightarrow$ do not change the type of the new IRR, this being respectively LRL and LRR for the top and bottom parts on the right of $\rightarrow\!\!\rightarrow$. The IRR's are also distinguished

by different pairs $(j_1, j_2)$. The type of an IGR is defined as the type of the IRR that it generates i.e. the type of its RHS. These together with their left-right reversed forms are all the different types of IGR's possible.

The corresponding right-left reversed results starting from an IRRP of type RLR also involve the parameters $j_1$ and $j_2$ obtained similarly but counting leftwards. Thus starting from

$$\mathtt{t}_1\mathtt{y}_n \ldots \underline{\mathtt{y}_1} \rightarrow\!\!\!\rightarrow \mathtt{t}_2\mathtt{z}_n \ldots \mathtt{z}_{1-} \tag{21}$$

likewise the following types of results are obtained which can be classified according to the leftmost position relative to $\mathtt{y}_1$, $(j_1)$ of the pointer. This satisfies $0 \leq j_1 \leq n-1$ and gives the following:

$$\mathtt{t}_1\mathtt{y}_n \ldots \underline{\mathtt{y}_1}\alpha \leftarrow \begin{cases} \mathtt{t}'_1\mathtt{y}_n \ldots \mathtt{y}_1\underline{\alpha'} \text{ for } j_1 = 0 \\ \mathtt{t}'_1\mathtt{y}_n \ldots \mathtt{y}_{j_1+2}\mathtt{y}'_{j_1+1} \ldots \mathtt{y}'_1\underline{\alpha'} \text{ for } 1 \leq j_1 \leq n-2 \\ \mathtt{t}'_1\underline{\mathtt{y}'_n}\mathtt{y}'_{n-1} \ldots \mathtt{y}'_1\alpha \text{ for } j_1 = n-1 \end{cases} \tag{22}$$

Naturally, (**??**) and (**??**) and are still valid and all the types of result in (**??**) have corresponding mirror image forms.

Simple examples of these are in (**??**), and (**??**) and (**??**) indicate the general method for deriving them which is as follows. After the symbol $\alpha$ has been added to the origins on the left, reverse steps of the TM are made recursively, making sure that all possible reverse steps at each stage are done and stopping only when further reverse steps are impossible without the knowledge of what the strings $\mathtt{T}_1$ and $\mathtt{T}_2$ are, as described in Section **??**.

These types of result in (**??**) are expressed with the shortest strings of symbols possible (i.e. the $\mathtt{y}$'s and $\mathtt{z}$'s). The strings $\mathtt{T}_1$ and $\mathtt{T}_2$ being arbitrary, so can be replaced by any strings. They do not have to have the same length. Thus an IGR is defined to have no redundant symbols where the pointer does not reach during its derivation. This is analogous to IRR's being irreducible. In the derivation of the IGR from an IRR of length $n$, the backward search to obtain the new origins and in the forward computation to obtain the new RHS, the pointer can obviously never move outside the strings of lengths $r_1$ and $r_2$ introduced above except for the last TM step in the forward computation. In addition all these positions of the pointer are reached during the derivation, the string of length $r_1$ for the derivation of a new origin and the string of length $r_2$ for the derivation of the new RHS.

If the pointer ends up at one end of the string $\mathtt{T}_2$ (indicated by $\underline{\mathtt{T}_2}$), the pointer position is clear from the context. The pair of strings of symbols $(\mathtt{T}_1, \mathtt{T}_2)$ of lengths $(n+1-r_1, n+1-r_2)$ respectively in (**??**) that are not passed by the pointer during the derivation of an IGR from an IRR of length $n$ that is the basis of its LHS will be removed and listed as "context pairs" so

that the result is presented in its minimal form i.e. as an IGR in computer output.

A IGR could be defined to include all the possible results that can be derived for any possible value of $\alpha$ (an IGR member), i.e. all the possible origins for each $\alpha$, but if there is not likely to be confusion I will refer to such statements as IGR's as was done above. Thus an IGR would be the union over $\alpha$ of the IGR members. An IGR member has the form (IRRP,$\alpha$) $\Rightarrow$ set of IRRP's, so the above results in (**??**) could be described as IGR members. Thus it would be possible for different RHS's of the IGR to have different values of $(r_1, r_2)$ corresponding to different values of $\alpha$, but these will be separated into different IGR's in the computer output.

Here `F` was defined as applied to an IRR pattern. When `F` is applied to an IGR or sequence of IGR's `X`, it is applied to the IRR pattern which is the RHS of `X`, so the RHS of `X` becomes the LHS of the new IGR `Y` derived from it by `F`. This shows that `F` derives from an IGR or a sequence of IGR's `X`, another IGR `Y` such that $\mathtt{X} \cdot \mathtt{Y}$ is a sequence of IGR's.

## 4.1 Computer representation and algorithms

There can be a problem that occurs in the computer representation of the IGR's after the context strings have been separated out, which is to determine whether the original IRRP on its left is of type LRL or RLR. Provided $n > \mathtt{1}$, it is not immediately obvious which is the case because the pointer positions and the parameter $j_1$ can be counted going either way, for example compare (**??**) with (**??**). The way it works is that a CS in the computer program output is represented as $\mathtt{CS}(\mathtt{t}, \mathtt{p}, \mathtt{l}, \mathtt{string})$ where `t` is the machine state, `p` is the pointer position counted from the left and is one for the symbol on the left, and is 0 for the position just to the left of this symbol, and is $\mathtt{l}+1$ for the position just to the right of the string, where $\mathtt{l} = n$ is the length of the string. The string is spelled out inside quote marks in printed output. After the context strings have been split out of the derived IGR, the pointer position in the origin of the IRRP set on the LHS of (**??**) is 1 by convention if the original IRRP (see (**??**)) (the LHS of the new IGR) was of type LRL or LRR because the pointer starts at 1 and is not affected by the truncation of the symbols from the right. If the original IRRP was of type RLR or RLL, the pointer position in its origin (LHS of (**??**)) is initially by convention at $n$ (i.e. the right hand end) and is reduced as a result of splitting out the context symbols. This for $j_1 = 0$ is position $n$ minus the length of the string of symbols removed also $n$ i.e. 0, and is $n$ minus the length of $\mathtt{y}_n \ldots \mathtt{y}_{j_1+2}$ otherwise, which is $j_1 + 1$. This value can never be 1, so

*$j_1 = \mathtt{1}$ is characteristic of the original IGR being of type LRL. This implies that the value* $\mathtt{p} = 1$ *in an origin CS on the LHS of an IGR in computer output*

*indicates that the context strings (*$\mathtt{T}_1$ *and* $\mathtt{T}_2$*) are added on the right, and on the left otherwise.*

For the case $n = 1$ this is obvious from the RHS of the IRRP on the LHS of the IGR which is of the form $\mathtt{t}_2\_\mathtt{z}_1$ or $\mathtt{t}_2\mathtt{z}_1\_$ according to whether the IGR is of type LRL or RLR respectively. This shows that this obvious convention for defining the pointer positions in the different cases distinguishes the LRL, LRR from the RLR, RLL types of IGR.

The above argument shows, when combined with Theorem **??**, that
(1) every IRR of length $n + 1$ of type LRL can be derived by `F` from another IRR of length $n$ of type LRL by an IGR with parameters $(r_1, r_2)$ of type (**??**).1 (LRL (**??**)) in satisfying $1 \leq r_1 \leq n$ and $1 \leq r_2 \leq n + 1$ as described and
(2) every IRR of length $n + 1$ of type LRR can be derived by `F` from another IRR of length $n$ of type LRL by an IGR of type (**??**).2 (LRR in (**??**)) (with parameters $r_1$ and $r_2$ such that $1 \leq r_1 \leq n$ and $r_2 = n + 1$.

These can be applied recursively to show that

**Theorem 4.1.** *any extendable IRR (type LRL or RLR) of length* $\geq 3$ *can be obtained from a member of IRR(2) by a sequence of substitutions of IGR's as described here under case (1). Any non-extendable IRR (type RLL or LRR) can be obtained from a member of IRR(2) by the above substitutions (0 or more) followed by a single substitution step under case (2).*

This theorem is illustrated by the example at the beginning of this section. This suggests the obvious process for generating the set of all the IGR's could start as follows after finding all the members of IRR(2). Essentially this was the method used in the latest version of the program [**?**] to generate Table **??**.

- Find all the members of IRR(2) and the IGR's used to generate them from the single TM steps.

- Likewise the IRR(3) can be obtained from the IRR(2) and the IGR's summarising this can be added while not duplicating any IGR's already found etc..

- This can be repeated to generate up to the IRR($n$).

After a while hopefully to generate the IRR($n + 1$) from the IRR($n$) will not require any IGR's that have not already been obtained for $n$ sufficiently large.

The following example was studied because the results from (**??**) became

very complicated.

$$\begin{array}{l} 1\underline{a} \rightarrow 2b\_ \\ 1\underline{b} \rightarrow 3\_b \\ 1\underline{c} \rightarrow 1b\_ \\ 2\underline{a} \rightarrow 3b\_ \\ 2\underline{b} \rightarrow 2c\_ \\ 2\underline{c} \rightarrow 1\_c \\ 3\underline{a} \rightarrow 1\_a \\ 3\underline{b} \rightarrow 1\_a \\ 3\underline{c} \rightarrow 3c\_ \end{array} \tag{23}$$

The results for the IGR's from TM **??** were as follows in Table **??** giving the maximum length of the computation rules as 10.

Table 1: IGR's generated by the computer program

| | |
|---|---|
| 1 | $1\underline{T_1} \to\to 1\_T_2 \overset{b}{\Rightarrow} 1\underline{c}T_1 \to\to 3\_bT_2$ |
| 2 | $1\underline{c}aT_1 \to\to 1\_T_2 \overset{b}{\Rightarrow} \left.\begin{matrix}2\underline{a}ca\\2\underline{a}cb\end{matrix}\right\}T_1 \to\to 3\_bT_2$ |
| 3 | $1\underline{c}aaT_1 \to\to 1\_T_2 \overset{b}{\Rightarrow} \left.\begin{matrix}1\underline{a}baa\\1\underline{a}bab\end{matrix}\right\}T_1 \to\to 3\_bT_2$ |
| 4 | $1\underline{c}ababT_1 \to\to 1\_T_2 \overset{b}{\Rightarrow} 1\underline{a}bbccbT_1 \to\to 3\_bT_2$ |
| 5 | $1\underline{c}ababcT_1 \to\to 1\_T_2 \overset{b}{\Rightarrow} 1\underline{a}bbccacT_1 \to\to 3\_bT_2$ |
| 6 | $1\underline{c}abcT_1 \to\to 1\_T_2 \overset{b}{\Rightarrow} \left.\begin{matrix}2\underline{a}ccac\\2\underline{a}ccbc\end{matrix}\right\}T_1 \to\to 3\_bT_2$ |
| 7 | $1\underline{c}cT_1 \to\to 1\_T_2 \overset{b}{\Rightarrow} 1\underline{a}bcT_1 \to\to 3\_bT_2$ |
| 8 | $2\underline{T_1} \to\to 1\_T_2 \overset{b}{\Rightarrow} 1\underline{a}T_1 \to\to 3\_bT_2$ |
| 9 | $3\underline{T_1} \to\to 1\_T_2 \overset{b}{\Rightarrow} 2\underline{a}T_1 \to\to 3\_bT_2$ |
| 10 | $3\underline{T_1} \to\to 1\_aT_2 \overset{c}{\Rightarrow} 3\underline{c}T_1 \to\to 2bb\underline{T_2}$ |
| 11 | $1\underline{c}cT_1 \to\to 1\_ababT_2 \overset{c}{\Rightarrow} 2\underline{b}bcT_1 \to\to 3bbbbb\underline{T_2}$ |
| 12 | $1\underline{c}aT_1 \to\to 1\_ababaT_2 \overset{c}{\Rightarrow} \left.\begin{matrix}3\underline{c}ca\\3\underline{c}cb\end{matrix}\right\}T_1 \to\to 3\_bababaT_2$ |
| 13 | $1\underline{c}aaT_1 \to\to 1\_ababaT_2 \overset{c}{\Rightarrow} \left.\begin{matrix}2\underline{b}baa\\2\underline{b}bab\end{matrix}\right\}T_1 \to\to 3\_bababaT_2$ |
| 14 | $1\underline{c}ababT_1 \to\to 1\_ababaT_2 \overset{c}{\Rightarrow} 2\underline{b}bbccbT_1 \to\to 3\_bababaT_2$ |
| 15 | $1\underline{c}ababcT_1 \to\to 1\_ababaT_2 \overset{c}{\Rightarrow} 2\underline{b}bbccacT_1 \to\to 3\_bababaT_2$ |
| 16 | $1\underline{c}abcT_1 \to\to 1\_ababaT_2 \overset{c}{\Rightarrow} \left.\begin{matrix}3\underline{c}ccac\\3\underline{c}ccbc\end{matrix}\right\}T_1 \to\to 3\_bababaT_2$ |
| 17 | $1\underline{c}cT_1 \to\to 1\_ababaT_2 \overset{c}{\Rightarrow} 2\underline{b}bcT_1 \to\to 3\_bababaT_2$ |
| 18 | $2\underline{T_1} \to\to 1\_ababaT_2 \overset{c}{\Rightarrow} 2\underline{b}T_1 \to\to 3\_bababaT_2$ |
| 19 | $1\underline{c}cT_1 \to\to 1\_ababcT_2 \overset{c}{\Rightarrow} 2\underline{b}bcT_1 \to\to 3bbbbbc\underline{T_2}$ |
| 20 | $1\underline{c}aT_1 \to\to 1\_abcT_2 \overset{c}{\Rightarrow} \left.\begin{matrix}3\underline{c}ca\\3\underline{c}cb\end{matrix}\right\}T_1 \to\to 1bbbb\underline{T_2}$ |
| 21 | $2\underline{T_1} \to\to 1\_cT_2 \overset{c}{\Rightarrow} 2\underline{b}T_1 \to\to 1bb\underline{T_2}$ |
| 22 | $1\underline{T_1} \to\to 3\_T_2 \overset{b}{\Rightarrow} 1\underline{c}T_1 \to\to 1\_aT_2$ |
| 23 | $1\underline{c}aT_1 \to\to 3\_T_2 \overset{b}{\Rightarrow} \left.\begin{matrix}2\underline{a}ca\\2\underline{a}cb\end{matrix}\right\}T_1 \to\to 1\_aT_2$ |
| 24 | $1\underline{c}aaT_1 \to\to 3\_T_2 \overset{b}{\Rightarrow} \left.\begin{matrix}1\underline{a}baa\\1\underline{a}bab\end{matrix}\right\}T_1 \to\to 1\_aT_2$ |
| 25 | $1\underline{c}ababT_1 \to\to 3\_T_2 \overset{b}{\Rightarrow} 1\underline{a}bbccbT_1 \to\to 1\_aT_2$ |
| 26 | $1\underline{c}ababcT_1 \to\to 3\_T_2 \overset{b}{\Rightarrow} 1\underline{a}bbccacT_1 \to\to 1\_aT_2$ |
| 27 | $1\underline{c}abcT_1 \to\to 3\_T_2 \overset{b}{\Rightarrow} \left.\begin{matrix}2\underline{a}ccac\\2\underline{a}ccbc\end{matrix}\right\}T_1 \to\to 1\_aT_2$ |
| 28 | $1\underline{c}cT_1 \to\to 3\_T_2 \overset{b}{\Rightarrow} 1\underline{a}bcT_1 \to\to 1\_aT_2$ |
| 29 | $2\underline{T_1} \to\to 3\_T_2 \overset{b}{\Rightarrow} 1\underline{a}T_1 \to\to 1\_aT_2$ |
| 30 | $3\underline{T_1} \to\to 3\_T_2 \overset{b}{\Rightarrow} 2\underline{a}T_1 \to\to 1\_aT_2$ |
| 31 | $2\underline{T_1} \to\to 3\_baT_2 \overset{c}{\Rightarrow} 2\underline{b}T_1 \to\to 3bbb\underline{T_2}$ |
| 32 | $1\underline{c}aT_1 \to\to 3\_babT_2 \overset{c}{\Rightarrow} \left.\begin{matrix}3\underline{c}ca\\3\underline{c}cb\end{matrix}\right\}T_1 \to\to 3\_babaT_2$ |
| 33 | $1\underline{c}aaT_1 \to\to 3\_babT_2 \overset{c}{\Rightarrow} \left.\begin{matrix}2\underline{b}baa\\2\underline{b}bab\end{matrix}\right\}T_1 \to\to 3\_babaT_2$ |
| 34 | $1\underline{c}ababT_1 \to\to 3\_babT_2 \overset{c}{\Rightarrow} 2\underline{b}bbccbT_1 \to\to 3\_babaT_2$ |

35 $1\underline{c}ababcT_1 \rightarrow\rightarrow 3\_babT_2 \overset{c}{\Rightarrow} 2\underline{b}bbccacT_1 \rightarrow\rightarrow 3\_babaT_2$

36 $1\underline{c}abcT_1 \rightarrow\rightarrow 3\_babT_2 \overset{c}{\Rightarrow} \left.\begin{matrix} 3\underline{c}ccac \\ 3\underline{c}ccbc \end{matrix}\right\} T_1 \rightarrow\rightarrow 3\_babaT_2$

37 $1\underline{c}cT_1 \rightarrow\rightarrow 3\_babT_2 \overset{c}{\Rightarrow} 2\underline{b}bcT_1 \rightarrow\rightarrow 3\_babaT_2$

38 $2\underline{T_1} \rightarrow\rightarrow 3\_babT_2 \overset{c}{\Rightarrow} 2\underline{b}T_1 \rightarrow\rightarrow 3\_babaT_2$

39 $3\underline{T_1} \rightarrow\rightarrow 3\_babT_2 \overset{c}{\Rightarrow} 3\underline{c}T_1 \rightarrow\rightarrow 3\_babaT_2$

40 $1\underline{T_1} \rightarrow\rightarrow 1T_2\_\left\{ \begin{matrix} \overset{a}{\Rightarrow} \left.\begin{matrix} 3T_1\underline{a} \\ 3T_1\underline{b} \end{matrix}\right\} \rightarrow\rightarrow 2T_2b\_ \\ \overset{c}{\Rightarrow} 2T_1\underline{c} \rightarrow\rightarrow 1T_2b\_ \end{matrix}\right.$

41 $1\underline{T_1} \rightarrow\rightarrow 2T_2\_ \overset{a}{\Rightarrow} \left.\begin{matrix} 3T_1\underline{a} \\ 3T_1\underline{b} \end{matrix}\right\} \rightarrow\rightarrow 3T_2b\_$

42 $3\underline{T_1} \rightarrow\rightarrow 2T_2\_ \overset{b}{\Rightarrow} 1T_1\underline{b} \rightarrow\rightarrow 2T_2c\_$

43 $1\underline{T_1} \rightarrow\rightarrow 3T_2\_ \overset{c}{\Rightarrow} 2T_1\underline{c} \rightarrow\rightarrow 3T_2c\_$

44 $3T_1bb\underline{b} \rightarrow\rightarrow 3T_2\_ \overset{c}{\Rightarrow} \left.\begin{matrix} 2T_1aab\underline{c} \\ 2T_1abb\underline{c} \end{matrix}\right\} \rightarrow\rightarrow 3T_2c\_$

45 $1\underline{T_1} \rightarrow\rightarrow 2T_2b\_ \overset{c}{\Rightarrow} 2T_1\underline{c} \rightarrow\rightarrow 3\underline{T_2}bc$

46 $1\underline{T_1} \rightarrow\rightarrow 2T_2c\_ \overset{c}{\Rightarrow} 2T_1\underline{c} \rightarrow\rightarrow 1T_2bb\_$

47 $3\underline{T_1} \rightarrow\rightarrow 3T_2c\_ \overset{b}{\Rightarrow} 1T_1\underline{b} \rightarrow\rightarrow 2T_2bb\_$

48 $1\underline{T_1} \rightarrow\rightarrow 2T_2bb\_ \overset{c}{\Rightarrow} 2T_1\underline{c} \rightarrow\rightarrow 1\underline{T_2}abc$

49 $3\underline{T_1} \rightarrow\rightarrow 3T_2bb\_ \overset{b}{\Rightarrow} 1T_1\underline{b} \rightarrow\rightarrow 1\underline{T_2}aba$

50 $3\underline{T_1} \rightarrow\rightarrow 3T_2cb\_ \overset{b}{\Rightarrow} 1T_1\underline{b} \rightarrow\rightarrow 3T_2bbb\_$

51 $3T_1bb\underline{b} \rightarrow\rightarrow 3T_2cb\_ \overset{a}{\Rightarrow} \left.\begin{matrix} 3T_1aab\underline{a} \\ 3T_1abb\underline{a} \\ 3T_1aab\underline{b} \\ 3T_1abb\underline{b} \end{matrix}\right\} \rightarrow\rightarrow 3T_2bbb\_$

52 $3\underline{T_1} \rightarrow\rightarrow 3T_2bbb\_ \overset{b}{\Rightarrow} 1T_1\underline{b} \rightarrow\rightarrow 3\underline{T_2}baba$

53 $1\underline{T_1} \rightarrow\rightarrow 3T_2bbbbb\_ \overset{a}{\Rightarrow} \left.\begin{matrix} 3T_1\underline{a} \\ 3T_1\underline{b} \end{matrix}\right\} \rightarrow\rightarrow 3\underline{T_2}bababa$

54 $3\underline{T_1} \rightarrow\rightarrow 3T_2bbbbb\_ \overset{b}{\Rightarrow} 1T_1\underline{b} \rightarrow\rightarrow 3\underline{T_2}bababa$

55 $3T_1bb\underline{b} \rightarrow\rightarrow 3T_2bbbbb\_ \overset{a}{\Rightarrow} \left.\begin{matrix} 3T_1aab\underline{a} \\ 3T_1abb\underline{a} \\ 3T_1aab\underline{b} \\ 3T_1abb\underline{b} \end{matrix}\right\} \rightarrow\rightarrow 3\underline{T_2}bababa$

Theorem **??** demonstrates the importance of derivations of IRR's using chains of IGR's substituted into each other. Connected with this is the relation 'can be followed by' which restricts the possible sequences of substitutions of IGR's. This is given in Table **??** and requires a match on the LHS and on the RHS in which the machine state and the symbol strings must match, as well as the direction for adding $\alpha$, and the first IGR must be of extendable type i.e. it must generate IRR's of type LRL or RLR. On the RHS of Table **??** (to the right of $\rightarrow$) all parts and sub-parts of an IGR referenced are included. Every IGR on the left of $\rightarrow$ can be followed by any IGR on the right of $\rightarrow$ in the same row.

In Table **??** and later in the paper, the IGR's from TM **??** will be named by the numbers in Table **??**. A part or a sub-part of an IGR will be referred to by following that number by a letter that refers to the part of the IGR associated with that letter which is the symbol above $\Rightarrow$ i.e. the symbol

called $\alpha$, possibly followed again by a number that determines the position of the CS in the list of CS's comprising the origin of the RHS of the IGR. For example the IGR $\mathtt{1}\underline{\mathtt{T_1}} \rightarrow\rightarrow \mathtt{1T_2\_} \overset{\mathtt{a}}{\Rightarrow} \mathtt{3T_1}\underline{\mathtt{b}} \rightarrow\rightarrow \mathtt{2T_2b\_}$ is IGR 40a2.

Table 2: The relation 'can be followed by'

| | | |
|---|---|---|
| 1 → | | 22, 23, 24, 25, 26, 27, 28, 32, 33, 34, 35, 36, 37 |
| 3b1, 3b2, 4, 5, 7, 8 → | | 22 |
| 2b1, 2b2, 6b1, 6b2, 9, 13c1, 13c2, 14, 15, 17, 18, 33c1, 33c2, 34, 35, 37, 38 | → | 29, 31, 38 |
| 12c1, 12c2, 16c1, 16c2, 36c1, 36c2, 39 → | | 30, 39 |
| 22 → | | 1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 13, 14, 15, 16, 18, 19 |
| 23b1, 23b2, 27b1, 27b2, 30 → | | 8, 18 |
| 24b1, 24b2, 25, 26, 28, 29 → | | 1 |
| 32c1, 32c2 → | | 29, 38 |
| 40a1, 40a2 → | | 42 |
| 41a1 → | | 49, 50, 52, 54 |
| 41a2 → | | 44, 49, 50, 51, 52, 54, 55 |
| 42 → | | 41, 46 |
| 47 → | | 41, 45, 48 |
| 50 → | | 43, 53 |
| 51a1, 51a2, 51a3 → | | 49, 52, 54 |
| 51a4 → | | 44, 49, 52, 54, 55 |

By examining these IGR's in Table **??** and the compatibility relations in Table **??** the following facts become evident:

1. There are a relatively small number of distinct origins of the LHS's of these IGR's. Each of these together with the value of $\alpha$ necessarily gives rise to the same origin of the RHS of the IGR regardless of the RHS of the LHS of the IGR. For example $\mathtt{1}\underline{\mathtt{c}}\mathtt{aT_1}$ with $\alpha = \mathtt{b}$ is always associated with $\left.\begin{matrix}\mathtt{2}\underline{\mathtt{a}}\mathtt{ca}\\ \mathtt{2}\underline{\mathtt{a}}\mathtt{cb}\end{matrix}\right\}\mathtt{T_1}$ in IGR's 2 and 23. Thus the presentation in Table **??** is far from optimal.

2. The left and right hand halves of the RHS of each IGR can be derived independently (it is only $\alpha$ that connects them).

3. IGR's can be chained together by $\cdot$ using substitutions for the arbitrary strings $\mathtt{T_1}$ and $\mathtt{T_2}$.

4. Sequence restrictions other than those coming from Table **??** may result from the way in which the sequences of substitutions operate.

5. By carrying out F to the RHS of an IGR, it is sometimes possible to deduce that no previous IGR's to a sequence of them can affect which IGR's can follow the sequence.

6. If by carrying out F to any sequence of IGR's to find which IGR's can be next in the sequence, there always results IGR's that have already been listed, then it would show that the set of IGR's found is sufficient to generate all the IRR's for the TM.

7. The starting points of these chains of IGR's to generate all the IRR's must be the IGR's involved in the IRR(2) together with their associated contexts. These are given in Table **??**.

Table 3: The IGR's and contexts that give on their RHS's all the IRR(2)

| IGR | Set of context pairs |
|---|---|
| 41 | $(\mathtt{a},\mathtt{b})$ |
| 22 | $(\mathtt{b},\mathtt{b})$ |
| 40 | $(\mathtt{c},\mathtt{b})$ |
| 8 | $(\mathtt{c},\mathtt{c})$ |
| 9 | $(\mathtt{a},\mathtt{a}),(\mathtt{b},\mathtt{a})$ |
| 47 | $(\mathtt{c},)$ |
| 45 | $(\mathtt{a},)$ |
| 21 | $(\mathtt{c},)$ |
| 10 | $(\mathtt{b},),(\mathtt{a},)$ |

Exploring sequences of IGR's chained together with $\cdot$ and applying $\mathtt{F}$ to them to generate new IGR's, was started beginning with IGR 22 because from Table **??**, IGR 22 clearly plays an important role. Applying $\mathtt{F}$ to IGR 22 in the usual way, doing the backward search stopping whenever the pointer gets to an end or next to an arbitrary string gives $1\alpha\underline{\mathtt{c}}\mathtt{T}_1 \overset{\mathtt{b}}{\leftarrow} 1\underline{\mathtt{c}}\mathtt{c}\mathtt{T}_1$ and $1\underline{\mathtt{b}}\mathtt{a}\mathtt{T}_2 \rightarrow 3\_\mathtt{b}\mathtt{a}\mathtt{T}_2$ so $1\underline{\mathtt{c}}\mathtt{T}_1 \rightarrow\!\!\rightarrow 1\_\mathtt{a}\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 1\underline{\mathtt{c}}\mathtt{c}\mathtt{T}_1 \rightarrow\!\!\rightarrow 3\_\mathtt{b}\mathtt{a}\mathtt{T}_2$. Removing the symbols that are not used gives the shortest form (the IGR produced) as $1\underline{\mathtt{T}_1} \rightarrow\!\!\rightarrow 1\_\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 1\underline{\mathtt{a}}\mathtt{T}_1 \rightarrow\!\!\rightarrow 3\_\mathtt{b}\mathtt{T}_2$ which is IGR 1. Coincidentally, IGR 1 is the only IGR that can precede IGR 22.

Next consider IGR 1 followed by IGR 22 denoted by $1\cdot 22$. In general fewer possibilities will result for the IGR's produced by $\mathtt{F}$ compared with $\mathtt{F}$ applied to 22 alone. The sequence of IGR's $1\cdot 22$ is $1\underline{\mathtt{T}_1} \rightarrow\!\!\rightarrow 1\_\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 1\underline{\mathtt{c}}\mathtt{T}_1 \rightarrow\!\!\rightarrow 3\_\mathtt{b}\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 1\underline{\mathtt{c}}\mathtt{c}\mathtt{T}_1 \rightarrow\!\!\rightarrow 1\_\mathtt{a}\mathtt{b}\mathtt{T}_2$ obtained by substituting $\mathtt{c}\mathtt{T}_1$ for $\mathtt{T}_1$ and $\mathtt{b}\mathtt{T}_2$ for $\mathtt{T}_2$ in IGR 22. The result of this is a composite IGR. By trying to apply $\mathtt{F}$ to this general form, results dependent on the arbitrary strings $\mathtt{T}_1$ and $\mathtt{T}_2$ will be produced. This starts by considering what CS's can lead to $1\alpha\underline{\mathtt{c}}\mathtt{c}\mathtt{T}_1$ for any symbol $\alpha$. It is easy to see that

$$1\alpha\underline{\mathtt{c}}\mathtt{c}\mathtt{T}_1 \begin{cases} \overset{\alpha=\mathtt{b}}{\leftarrow} 1\underline{\mathtt{c}}\mathtt{c}\mathtt{c}\mathtt{T}_1 \\ \leftarrow 2\alpha\mathtt{c}\underline{\mathtt{c}}\mathtt{T}_1 \end{cases}. \tag{24}$$

in one TM step in either case. The first of these will lead to the IRRP $1\underline{\mathtt{c}}\mathtt{c}\mathtt{c}\mathtt{T}_1 \rightarrow\!\!\rightarrow 3\_\mathtt{b}\mathtt{a}\mathtt{b}\mathtt{T}_2$ because $1\underline{\mathtt{b}}\mathtt{a}\mathtt{b}\mathtt{T}_2 \rightarrow 3\_\mathtt{b}\mathtt{a}\mathtt{b}\mathtt{T}_2$. The strings $\mathtt{c}\mathtt{c}\mathtt{T}_1$ and $\mathtt{a}\mathtt{b}\mathtt{T}_2$ are not changed because the pointer does not enter them in the derivation of the IRRP, so the IGR used is $1\underline{\mathtt{T}_1} \rightarrow\!\!\rightarrow 1\_\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 1\underline{\mathtt{c}}\mathtt{T}_1 \rightarrow\!\!\rightarrow 3\_\mathtt{b}\mathtt{T}_2$ i.e. IGR 1 in Table **??**. This same result has been obtained because it is the special case of the above result where the first symbols of $\mathtt{T}_1$ and $\mathtt{T}_2$ are $\mathtt{c}$ and $\mathtt{b}$ respectively.

The second result of (**??**) has reached condition 2 in the backward search if $\mathtt{T}_1$ is the empty string, so the the backward search cannot be continued further in that case.

If $\mathtt{T}_1$ is not the empty string, the general reasoning indicates that $\mathtt{T}_1$ needs to be specialised further by prepending the sequence of IGR's $\mathtt{1 \cdot 22}$ with others, however the backward search can be logically continued giving

$$2\alpha\mathtt{c}\underline{\mathtt{c}}\mathtt{T}_1 \leftarrow 2\alpha\underline{\mathtt{b}}\mathtt{c}\mathtt{T}_1 \begin{cases} \overset{\alpha=\mathtt{b}}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{b}\mathtt{c}\mathtt{T}_1 \\ \overset{\alpha=\mathtt{c}}{\leftarrow} 2\underline{\mathtt{b}}\mathtt{b}\mathtt{c}\mathtt{T}_1 \end{cases} \tag{25}$$

which is independent of $\mathtt{T}_1$ because the first of these reverse steps from $2\alpha\mathtt{c}\underline{\mathtt{c}}\mathtt{T}_1$ cannot lead to any other result than the one indicated (because there is no TM step ending in $\mathtt{2\_}\beta$ no matter what the symbol $\beta$ in $\mathtt{T}_1$ is). This shows that if $\mathtt{T}_1$ is not the empty string, the result will always be that condition 1 is reached, giving another IGR. However this is an exceptional case, and the general method used for doing F to IRR's will be followed i.e. whenever the pointer reaches next to an arbitrary string the computation will stop, thus in this case only one IGR is generated together with an RCS. This makes the general procedure less complicated.

Returning to the general argument, taking a further step back in the sequence of IGR's to be considered gives $\mathtt{22 \cdot 1 \cdot 22}$. This sequence gives

$$1\underline{\mathtt{T}_1} \rightarrow\!\!\rightarrow \mathtt{3\_T}_2 \overset{22}{\Rightarrow} 1\underline{\mathtt{c}}\mathtt{T}_1 \rightarrow\!\!\rightarrow \mathtt{1\_aT}_2 \overset{1\cdot 22}{\Rightarrow} 1\underline{\mathtt{c}}\mathtt{ccT}_1 \rightarrow\!\!\rightarrow \mathtt{1\_abaT}_2 \tag{26}$$

the second part of which comes from $\mathtt{1 \cdot 22}$ above. The symbols above the symbols $\Rightarrow$ respectively indicate IGR 22 (with $\alpha = \mathtt{b}$) and as above $\mathtt{1 \cdot 22}$ (with two steps of IGR's with $\alpha = \mathtt{b}$). Applying F to this starts by the backward search from $1\alpha\underline{\mathtt{c}}\mathtt{ccT}_1$ giving (e.g. using (**??**) which was used one step later than when it was obtained) the following

$$1\alpha\underline{\mathtt{c}}\mathtt{ccT}_1 \begin{cases} \overset{\mathtt{b}}{\leftarrow} 1\underline{\mathtt{c}}\mathtt{cccT}_1 \\ \leftarrow 2\alpha\mathtt{c}\underline{\mathtt{c}}\mathtt{ccT}_1 \leftarrow 2\alpha\underline{\mathtt{b}}\mathtt{ccT}_1 \begin{cases} \overset{\mathtt{b}}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{bccT}_1. \\ \overset{\mathtt{c}}{\leftarrow} 2\underline{\mathtt{b}}\mathtt{bccT}_1 \end{cases} \end{cases} \tag{27}$$

Combining this with $\mathtt{1}\underline{\mathtt{b}}\mathtt{abaT}_2 \rightarrow \mathtt{3\_babaT}_2$ and $\mathtt{1}\underline{\mathtt{c}}\mathtt{abaT}_2 \rightarrow \mathtt{3bbcb}\underline{\mathtt{T}_2}$ and absorbing any unchanged symbols into $\mathtt{T}_1$ or $\mathtt{T}_2$ because the pointer has not reached them gives the results 1 again, IGR 7 and

$$1\underline{\mathtt{c}}\mathtt{cT}_1 \rightarrow\!\!\rightarrow \mathtt{1\_abaT}_2 \overset{\mathtt{c}}{\Rightarrow} 2\underline{\mathtt{b}}\mathtt{bcT}_1 \rightarrow\!\!\rightarrow \mathtt{3bbcb}\underline{\mathtt{T}_2}. \tag{28}$$

which is not in Table **??**. Actually IGR 17 is a special case of IGR 11 which is itself a special case of (**??**) (only on the LHS's), formed by successively decreasing the length of a string in the RHS by 1. Because in these three cases, IGR 17 uniquely has the pointer finishing at the $\alpha$ end of the string in

RHS of the RHS, such a result as IGR 17 cannot be continued by specialising $\mathtt{T}_2$ and continuing the computation to the end in the RHS, so IGR 17 is in some sense a completion of (**??**) and IGR 11.

In (**??**) because of the absence of a branch of the backward search taking the pointer to the opposite end of the string from $\alpha$ i.e. there are no RCS's, any special cases of $\mathtt{T}_1$ that would result from prior IGR's in the sequence could not affect the new origins of IGR's that could be next in the sequence, only the RHS's could vary. This is because the general form of the derivation of a new origin follows the pattern in (**??**) whatever substitutes for $\mathtt{T}_1$. Because 1 can also be preceded by 24b1, 24b2, 25, 26, 28, 29, these cases could now be considered in turn preceding $1 \cdot 22$.

What has started to be explored above is that there could be an alternative algorithm to generate the IGR's directly from each other by applying F in these general cases for arbitrary strings $\mathtt{T}_1$ and $\mathtt{T}_2$ starting from the IGR's needed to derive the IRR(2) from the single TM steps. This is extremely complicated because of dealing with new origins and RHS's together, and there are a lot of different cases that can arise by applying F to sequences of IGR's determined by using the relation 'can be followed by' in Table **??**. Further attempts was made to do this. However this did not work well, because as above IGR's not in (**??**) were generated, and it was not clear how much context needs to be added at each stage. Also in the example above as a result of the absence of RCS's, the derivation of the new origins is halted but that of RHS's is not. This suggests treating the backward and forward derivations in IGR's more separately i.e. considering at first just backward searches to get the new origins for the IGR's. For this I introduced the new concept of LIGR (Left IGR) because the RHS's can presumably be filled in later just with forward computations.

# 5 Introducing LIGR's as an aid to finding the IGR's for a TM

The aim of this section will be to formulate this procedure precisely and to demonstrate it and show that if it does come to an end, then the LIGR's so obtained from a Turing Machine can form the basis of an alternative intuitive description of the action of the TM.

An LIGR or left IGR is the origin of the LHS of an IGR, and the origin of the RHS of the same IGR, combined with the symbol $\alpha$. For example IGR's 2, 23 have the common LIGR

$$1\alpha\underline{\mathtt{c}}\mathtt{a}\mathtt{T}_1 \overset{\alpha=\mathtt{b}}{\leftarrow} \left.\begin{matrix} 2\underline{\mathtt{a}}\mathtt{c}\mathtt{a} \\ 2\underline{\mathtt{a}}\mathtt{c}\mathtt{b} \end{matrix}\right\}\mathtt{T}_1. \tag{29}$$

Similarly

$$1\alpha\underline{c}aT_1 \overset{\alpha=c}{\leftarrow} \left.\begin{matrix} 3\underline{c}ca \\ 3\underline{c}cb \end{matrix}\right\} T_1 \tag{30}$$

is common to IGR's 12 and 20. These examples show that in common with IGR's, LIGR's can have parts (labelled by $\alpha$) and sub-parts that will be labelled in lexicographical order of the strings with the most significant symbol (sorted first) being where the pointer is. Also the symbol $\alpha$ may be omitted for brevity because it is always at the opposite end of the string from $T_1$, which in the context of LIGR's will be called just T because $T_2$ is not involved. For example if (??) and (??) are treated as parts of the complete LIGR X then (??) is X.b and (??) is X.c. The length of an LIGR will be the length of the symbol string on its right, which is one more than the length of the symbol string on the left assuming $\alpha$ is omitted. This notation with the reverse facing arrow will be used because as usual the arrows ($\leftarrow$ or $\rightarrow$) indicate the direction of the computation of the TM as distinct from direction of logical derivation indicated by $\Rightarrow$. Thus LIGR's are also reverse computation rules, but very special ones because they arise in the context of IGR's.

There are obvious advantages of treating IGR's in this way as can be seen in the drastically shortened list of results (12 LIGR's from Table ?? not counting parts and sub-parts separately). Moreover if the LHS of an LIGR matches the origin of an IRR, F applied to this IRR has as origins the result of the substitution for T in the RHS of the LIGR, and its RHS can be computed directly from the RHS of the IRR using alpha of the LIGR. Thus the RHS's can be filled in later and do not need to be recorded in the rule (an IGR) for generating new IRR's from existing ones.

There is however a limitation to completely ignoring the RHS's which is that LIGR's could be thought of (vacuous LIGR's) which can never be used in practice because they do not match any IRR's. The consequences of this will be followed up later in Section ??.

## 5.1 Associativity of composition

Underlying this is the assumption that the operation $\cdot$ is associative. This is such a basic result that it might never be questioned, being regarded as obvious. It means that for any LIGR's $L_1$,$L_2$ and $L_3$ the following is true:

$$(L_1 \cdot L_2) \cdot L_3 = L_1 \cdot (L_2 \cdot L_3) \tag{31}$$

which implies that for any string of LIGR's, the compositions can be done in any order provided the sequence of LIGR's is maintained e.g. $L_1 \cdot ((L_2 \cdot L_3) \cdot L_4) = (L_1 \cdot L_2) \cdot (L_3 \cdot L_4)$. Suppose $L_i = x_{i1}\underline{y_{i1}} \ldots y_{ir_i}T_i \leftarrow x_{i2}\underline{y'_{i1}} \ldots y'_{ir_i+1}T_i$ for $1 \leq i \leq 3$ (where $x_{i1}$ and $x_{i2}$ are states, y and y′ with subscripts are symbols and $\alpha$ is put on the left for the computations). To show this first note that the meaning of an LIGR is that the CS on the left can be replaced by the CS on

the right regardless of the arbitrary string denoted by $\mathtt{T}$ or $\mathtt{T}$ with a subscript. To apply an LIGR, the string $\mathtt{T}$ is chosen to match the given CS if possible. If it matches, it can be replaced by the corresponding string on the right with the same chosen substring $\mathtt{T}$.

Thus the composition of the three LIGR's written out in full as

$$\begin{array}{l} \mathtt{L}_1 = \mathtt{x}_{11}\underline{\mathtt{y}_{11}} \cdots \mathtt{y}_{1\mathtt{r}_1}\mathtt{T}_1 \leftarrow \mathtt{x}_{12}\underline{\mathtt{y}'_{11}} \cdots \mathtt{y}'_{1\mathtt{r}_1+1}\mathtt{T}_1 \\ \mathtt{L}_2 = \mathtt{x}_{21}\underline{\mathtt{y}_{21}} \cdots \mathtt{y}_{2\mathtt{r}_2}\mathtt{T}_2 \leftarrow \mathtt{x}_{22}\underline{\mathtt{y}'_{21}} \cdots \mathtt{y}'_{2\mathtt{r}_2+1}\mathtt{T}_2 \\ \mathtt{L}_3 = \mathtt{x}_{31}\underline{\mathtt{y}_{31}} \cdots \mathtt{y}_{3\mathtt{r}_3}\mathtt{T}_3 \leftarrow \mathtt{x}_{32}\underline{\mathtt{y}'_{31}} \cdots \mathtt{y}'_{3\mathtt{r}_3+1}\mathtt{T}_3 \end{array} \tag{32}$$

requires the matching of $\mathtt{L}_2$ and $\mathtt{L}_3$ by choosing $\mathtt{T}_2$ and $\mathtt{T}_3$ for the given $\mathtt{T}_1$ such that

$$\begin{array}{l} \mathtt{x}_{21} = \mathtt{x}_{12}, \underline{\mathtt{y}_{21}} \cdots \mathtt{y}_{2\mathtt{r}_2}\mathtt{T}_2 = \underline{\mathtt{y}'_{11}} \cdots \mathtt{y}'_{1\mathtt{r}_1+1}\mathtt{T}_1 \\ \mathtt{x}_{31} = \mathtt{x}_{22}, \underline{\mathtt{y}_{31}} \cdots \mathtt{y}_{3\mathtt{r}_3}\mathtt{T}_3 = \underline{\mathtt{y}'_{21}} \cdots \mathtt{y}'_{2\mathtt{r}_2+1}\mathtt{T}_2 \end{array} \tag{33}$$

Although there are 9 cases here because $\mathtt{r}_2$ can be greater, equal, or less than $\mathtt{r}_1{+}1$, and likewise for $\mathtt{r}_3$ and $\mathtt{r}_2{+}1$, (??) is the *only* condition needed regardless of the order of composition, therefore this order does not affect the result.

## 5.2 Sequences of LIGR's and F

A sequence of LIGR's is a chain of LIGR's that can follow each other written with $\cdot$ between them so for example if

$$\mathtt{L}_1 = 1\underline{\mathtt{c}}\mathtt{aT} \overset{\mathtt{b}}{\leftarrow} \left.\begin{array}{l} 2\underline{\mathtt{a}}\mathtt{ca} \\ 2\underline{\mathtt{a}}\mathtt{cb} \end{array}\right\}\mathtt{T}. \tag{34}$$

$$\mathtt{L}_2 = 2\underline{\mathtt{T}}\left\{\begin{array}{l} \overset{\mathtt{b}}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{T} \\ \overset{\mathtt{c}}{\leftarrow} 2\underline{\mathtt{b}}\mathtt{T} \end{array}\right. \tag{35}$$

$$\mathtt{L}_3 = 1\underline{\mathtt{T}} \overset{\mathtt{b}}{\leftarrow} 1\underline{\mathtt{c}}\mathtt{T} \tag{36}$$

then

$$\begin{array}{l} \mathtt{L}_{1.2} = 1\underline{\mathtt{c}}\mathtt{aT} \overset{\mathtt{b}}{\leftarrow} 2\underline{\mathtt{a}}\mathtt{cbT} \\ \mathtt{L}_{2.1} = 2\underline{\mathtt{T}} \overset{\mathtt{b}}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{T} \end{array} \tag{37}$$

and the chain $\mathtt{L}_{1.2} \cdot \mathtt{L}_{2.1} \cdot \mathtt{L}_3$ is the the result of the three substitutions of the LHS performed in that sequence giving

$$1\underline{\mathtt{c}}\mathtt{aT} \leftarrow 2\underline{\mathtt{a}}\mathtt{cbT} \leftarrow 1\underline{\mathtt{a}}\mathtt{acbT} \leftarrow 1\underline{\mathtt{c}}\mathtt{aacbT} \tag{38}$$

so the combination is $1\underline{\mathtt{c}}\mathtt{aT} \overset{\mathtt{bbb}}{\leftarrow} 1\underline{\mathtt{c}}\mathtt{aacbT}$.

Similarly to the way in which F was applied to IGR's, this can obviously be done for LIGR's by just taking the origins only i.e. the parts to the left of $\rightarrow\rightarrow$. The result of F applied to an LIGR of length 1 cannot give rise to any residual CS's because there is not enough "room" because there is only one

possible reverse TM step to $\alpha$ for example with $\mathtt{L}_{2.1}$, applying $\mathtt{F}$ starts with $\mathtt{1}\alpha\underline{\mathtt{a}}\mathtt{T} \overset{\alpha=\mathtt{b}}{\leftarrow} \mathtt{1}\underline{\mathtt{c}}\mathtt{aT}$ and after absorbing the $\mathtt{a}$ into the arbitrary string $\mathtt{T}$ because it is not involved gives $\mathtt{1}\underline{\mathtt{T}} \overset{\mathtt{b}}{\leftarrow} \mathtt{1}\underline{\mathtt{c}}\mathtt{T}$ where it unnecessary to write the $\alpha$ explicitly and often the superscripted $\alpha$ value will not be written. The effect of applying $\mathtt{F}$ to a sequence of LIGR's can be affected by adding an extra LIGR to the beginning of the sequence. Suppose an LIGR or a sequence of LIGR's $\mathtt{X}_1$ combined as above with $\alpha$ is of the form (putting the $\alpha$ in for clarity)

$$\mathtt{s}_1\alpha\underline{\mathtt{y}_1}\ldots\mathtt{y}_\mathtt{r}\mathtt{T}_1 \leftarrow \mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{\mathtt{r}+1}\mathtt{T}_1. \tag{39}$$

Likewise let $\mathtt{X}_2$ be

$$\mathtt{s}'_1\alpha'\underline{\mathtt{y}'_1}\ldots\mathtt{y}'_\mathtt{q}\mathtt{T}_2 \leftarrow \mathtt{s}'_2\underline{\mathtt{z}'_1}\ldots\mathtt{z}'_{\mathtt{q}+1}\mathtt{T}_2. \tag{40}$$

Then for the sequence $\mathtt{X}_2\cdot\mathtt{X}_1$ to be possible requires, $\mathtt{s}'_2 = \mathtt{s}_1$ and $\underline{\mathtt{y}_1}\ldots\mathtt{y}_\mathtt{r}$ is a substring of $\underline{\mathtt{z}'_1}\ldots\mathtt{z}'_{\mathtt{q}+1}$ on the left. The result of $\mathtt{X}_2\cdot\mathtt{X}_1$ is

$$\begin{array}{l}\mathtt{s}'_1\alpha\alpha'\underline{\mathtt{y}'_1}\ldots\mathtt{y}'_\mathtt{q}\mathtt{T}_2 \leftarrow \mathtt{s}'_2\alpha\underline{\mathtt{z}'_1}\ldots\mathtt{z}'_{\mathtt{q}+1}\mathtt{T}_2 = \\ \mathtt{s}_1\alpha\underline{\mathtt{y}_1}\ldots\mathtt{y}_\mathtt{r}\mathtt{z}'_{\mathtt{r}+1}\ldots\mathtt{z}'_{\mathtt{q}+1}\mathtt{T}_2 \leftarrow \mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{\mathtt{r}+1}\mathtt{z}'_{\mathtt{r}+1}\ldots\mathtt{z}'_{\mathtt{q}+1}\mathtt{T}_2.\end{array} \tag{41}$$

i.e.

$$\mathtt{s}'_1\alpha\alpha'\underline{\mathtt{y}'_1}\ldots\mathtt{y}'_\mathtt{q}\mathtt{T} \leftarrow \mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{\mathtt{r}+1}\mathtt{z}'_{\mathtt{r}+1}\ldots\mathtt{z}'_{\mathtt{q}+1}\mathtt{T}. \tag{42}$$

Comparing (**??**) with (**??**) shows that the effect on $\mathtt{X}_1$ of preceding it with $\mathtt{X}_2$ is to add extra symbols next to $\mathtt{T}$ in its right hand member. Therefore the results of the backward search starting from the RHS of (**??**) and shortened to the shortest form are reproduced when started from the RHS of (**??**) provided the pointer ends up at $\alpha$ when started from $\mathtt{X}_1$; these give rise to LIGR's. In addition there may be some extra LIGR's resulting from the pointer reaching the extra symbols which may be classified by the position of the rightmost symbol reached. Crucially, this happens only when $\mathtt{F}$ applied to $\mathtt{X}_1$ leads to cases in the backward search when the pointer ends up at the opposite end of the string from $\alpha$ i.e. condition 2 is reached. These were termed residual CS's (RCS's) because they are cases that do not lead directly to any more IGR's and LIGR's but indicate the possibility of them if the sequence of LIGR's to which $\mathtt{F}$ is applied increases in length as a result of a preceding LIGR appended to the sequence in question. Finally, there may be results of this where the pointer ends up at the opposite end of the string from $\alpha$ i.e. the pointer goes right when the rightmost symbol is reached, when starting from $\mathtt{X}_2\cdot\mathtt{X}_1$. These are the new RCS's in the result of $\mathtt{F}$ applied to (**??**) and will be designated as $\mathtt{F}_2(\mathtt{X}_2\cdot\mathtt{X}_1)$. Therefore it makes sense to introduce $\Delta\mathtt{F}_1$ as the set of extra LIGR's from $\mathtt{F}$, as a result of adding an extra LIGR $\mathtt{X}_2$ at the beginning of the sequence where $\mathtt{X}_1 = \mathtt{Y}_1\cdot\mathtt{Y}_2\ldots\cdot\mathtt{Y}_\mathtt{n}$ is a sequence of LIGR's:

$$\Delta\mathtt{F}_1(\mathtt{X}_2,\mathtt{X}_1) = \mathtt{F}_1(\mathtt{X}_2\cdot\mathtt{X}_1)\setminus\mathtt{F}_1(\mathtt{X}_1). \tag{43}$$

In this notation the main result of the preceding paragraph can be written as

$$\mathtt{F}_2(\mathtt{X}_1) = \emptyset \Rightarrow \Delta\mathtt{F}_1(\mathtt{X}_2, \mathtt{X}_1) = \emptyset. \tag{44}$$

where the result of $\mathtt{F}$ applied to a sequence of LIGR's was split into two components $\mathtt{F} = (\mathtt{F}_1, \mathtt{F}_2)$ where $\mathtt{F}_1$ is the set of LIGR's produced and $\mathtt{F}_2$ is the set of RCS's produced. The converse is not true because it could be that there are some reverse search paths that go beyond the symbols in $\mathtt{X}_1$ but none of them go back to $\alpha$. In this case $\mathtt{F}_2(\mathtt{X}_1) \neq \emptyset$ and $\Delta\mathtt{F}_1(\mathtt{X}_2, \mathtt{X}_1) = \emptyset$. The result of $\mathtt{F}$ for a collection of LIGR's in a sequence combined with $\cdot$ is defined as the result of $\mathtt{F}$ for the combined LIGR, which actually only depends on the its RHS and results from applying the backward search algorithm to it. A consequence of this is that the arguments of $\mathtt{F}$ can be written in different ways e.g. the sequence of LIGR's can be replaced by the string of symbols in the RHS of the combined LIGR.

## 5.3 Evaluating (??) one extra symbol at a time

The following is a description of the above calculation taken one symbol at a time. It can be applied when several symbols are added in one step from a single LIGR as was the original intention, or when a sequence of LIGR's is added that each contribute just one symbol etc..

The result of (**??**) can obviously be obtained by adding each symbol one at a time and accumulating the results for each extra symbol added. Suppose the extra symbols added to the starting CS $\mathtt{s}_2\underline{\mathtt{z}_1}\cdots\mathtt{z}_{r+1}$ from $\mathtt{X}_1$ as a result of preceding it with $\mathtt{X}_2$ are $\mathtt{z}'_{r+1}\ldots\mathtt{z}'_{q+1}$ as in (**??**) then one can write:

$$\begin{aligned}&\mathtt{F}_1(\mathtt{X}_2\cdot\mathtt{X}_1) = \mathtt{F}_1(\mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1}\mathtt{z}'_{r+1}\ldots\mathtt{z}'_{q+1})\\ &= \Delta\mathtt{F}_1(\mathtt{z}'_{r+1}, \mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1})\cup\\ &\textstyle\bigcup_{i=r+2}^{i=q+1}\Delta\mathtt{F}_1(\mathtt{z}'_i, \mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1}\mathtt{z}'_{r+1}\ldots\mathtt{z}'_{i-1})\cup\mathtt{F}_1(\mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1})\end{aligned} \tag{45}$$

where the union of all the $\Delta\mathtt{F}_1$ terms is (**??**) and last term is $\mathtt{F}_1(\mathtt{X}_1)$. Each step corresponding to one term in the multiple union uses the RCS's from the previous step. These RCS's with the single extra symbol are the starting points of the continuing backward search which would of course stop if at some point there were no more RCS's. In more detail, in order to calculate a typical term

$$\Delta\mathtt{F}_1(\mathtt{z}'_i, \mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1}\mathtt{z}'_{r+1}\ldots\mathtt{z}'_{i-1}) \tag{46}$$

which is by definition

$$\mathtt{F}_1(\mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1}\mathtt{z}'_{r+1}\ldots\mathtt{z}'_i)\setminus\mathtt{F}_1(\mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1}\mathtt{z}'_{r+1}\ldots\mathtt{z}'_{i-1}), \tag{47}$$

in the backward search, before ending up at $\alpha$, the pointer must first reach the symbol at $\mathtt{z}'_{i-1}$, therefore the backward search can start from there i.e.

$$\mathtt{F}_2(\mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1}\mathtt{z}'_{r+1}\ldots\mathtt{z}'_{i-1})\mathtt{z}'_i \tag{48}$$

where the last symbol is concatenated to the RCS given. If the pointer reaches $\alpha$ the result is a new LIGR otherwise it gives an RCS which is in

$$\mathtt{F}_2(\mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1}\mathtt{z}'_{r+1}\ldots\mathtt{z}'_i). \tag{49}$$

Putting $i = r+1$ initially, then this shows that the backward search starts from $\mathtt{F}_2(\mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1})\mathtt{z}'_{r+1}$ i.e. the set of RCS's from the initial backward search for $\mathtt{X}_1$ each appended with the first extra symbol $\mathtt{z}'_{r+1}$ on the right. If the pointer reaches $\alpha$ as the backward search continues, this gives a new LIGR otherwise if goes to the other end it gives an RCS in $\mathtt{F}_2(\mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1}\mathtt{z}'_{r+1})$. If the pointer comes to a point where the backward search can go no further or an infinite stationary loop no results are contributed. Then for $i = r+2$, the backward search starts from each of the above RCS's appended with $\mathtt{z}'_{r+2}$ on the right i.e. $\mathtt{F}_2(\mathtt{s}_2\underline{\mathtt{z}_1}\ldots\mathtt{z}_{r+1}\mathtt{z}'_{r+1})\mathtt{z}'_{r+2}$ and continues until either the pointer reaches $\alpha$ giving a new LIGR, or away from it giving an RCS etc. until $i = q+1$ is reached i.e. until all the new symbols have been added and all possible backward search paths are followed at each stage. If at any stage there are no RCS's in $\mathtt{F}_2$ it terminates. All the new LIGR's are accumulated and any final RCS's are noted. Naturally, there is an equivalent version of this if $\alpha$ is on the right.

# 6 The procedure for finding the LIGR's for a TM

This is extremely complicated and is very hard to describe so the reader is not expected to understand it immediately. For this reason I attempt to do so here in this section and then the procedure is applied to TM (**??**) in detail in Section **??** when it should become clearer. It is based on the following facts:
Every LIGR is part of an IGR.
Every IGR describes the result of F applied to an IRR.
Every IRR can be generated by a sequence of applications of F to a single TM step.
Therefore, because the backward searching on the left and the forward computations on the right are independent of each other, *every LIGR is the result of* F *applied to a sequence of LIGR's combined with* $\cdot$. This suggests a recursive algorithm that uses this starting from the LIGR's involved (the LIGR(2)) in getting the IRR(2) from the TM steps using F. A simple example of getting some members of LIGR(2) follows. Start from the single TM step $\mathtt{1}\underline{\mathtt{a}} \to \mathtt{2b}\_$. Because the TM moves right $\alpha$ is put on the right and the arbitrary strings $\mathtt{T}_1$ and $\mathtt{T}_2$ are on the left (it would be the other way round otherwise). Doing the backward search consists of just 1 step in each case. There are two options for

$\alpha = \mathtt{a}$ and one for $\alpha = \mathtt{c}$ as follows:

$$1\mathtt{T}_1\underline{\mathtt{a}}\alpha \begin{cases} \overset{\mathtt{a}}{\leftarrow} \begin{cases} 3\mathtt{T}_1\mathtt{a}\underline{\mathtt{a}} \\ 3\mathtt{T}_1\mathtt{a}\underline{\mathtt{b}} \end{cases} . \\ \overset{\mathtt{c}}{\leftarrow} 2\mathtt{T}_1\mathtt{a}\underline{\mathtt{c}} \end{cases} \tag{50}$$

The forward computations are just $2\mathtt{T}_2\mathtt{b}\underline{\mathtt{a}} \rightarrow 3\mathtt{T}_2\mathtt{bb}\_$ and $2\mathtt{T}_2\mathtt{b}\underline{\mathtt{c}} \rightarrow 3\underline{\mathtt{T}_2}\mathtt{bc}$. Putting this together as an IGR and simplifying as much as possible by redefining $\mathtt{T}_1$ and $\mathtt{T}_2$ to include *all* the arbitrary symbols in the strings that are not involved in the computation (in this case renaming the arbitrary strings $\mathtt{T}_1\mathtt{a}$ as $\mathtt{T}_1$ and $\mathtt{T}_2\mathtt{b}$ and $\mathtt{T}_2$), gives the following for $\alpha = \mathtt{a}$: $1\underline{\mathtt{T}_1} \rightarrow 2\mathtt{T}_2\_ \overset{\mathtt{a}}{\Rightarrow} 3\mathtt{T}_1\underline{\mathtt{d}} \rightarrow\!\!\rightarrow 3\mathtt{T}_2\mathtt{b}\_$ which is IGR 41 in Table **??** and corresponds to LIGR 7 in (**??**), and likewise for $\alpha = \mathtt{c}$, after renaming the arbitrary string $\mathtt{T}_1\mathtt{a}$ as $\mathtt{T}_1$ gives the following $1\underline{\mathtt{T}_1} \rightarrow 2\mathtt{T}_2\mathtt{b}\_ \overset{\mathtt{c}}{\Rightarrow} 2\mathtt{T}_1\underline{\mathtt{c}} \rightarrow\!\!\rightarrow 3\underline{\mathtt{T}_2}\mathtt{bc}$ which is IGR 45 in Table **??** and corresponds to LIGR 6 in (**??**). Note that for single TM steps regarded as IRR patterns there is no origin as distinct from its LHS so effectively $\rightarrow$ is the same as $\rightarrow\!\!\rightarrow$. Therefore in order to obtain the LIGR's directly without dealing with the RHS's, it is simply necessary to take (**??**) and directly rename the arbitrary string $\mathtt{T}_1\mathtt{a}$ as $\mathtt{T}$ because it is not altered by the computation and deduce the LIGR's as $1\underline{\mathtt{T}} \overset{\mathtt{a}}{\leftarrow} 3\mathtt{T}\begin{Bmatrix}\underline{\mathtt{a}} \\ \underline{\mathtt{b}}\end{Bmatrix}$ and $1\underline{\mathtt{T}} \overset{\mathtt{c}}{\leftarrow} 2\mathtt{T}\underline{\mathtt{c}}$.

**Algorithm 6.1.** *After finding the set LIGR(2) as above, the LIGR's involved in forming the IRR*$(n+1)$ *from the IRR*$(n)$ *need to be found for all* $n \geq 2$. *This can be done by finding, for each sequence of LIGR's that can be substituted into each other in sequence as in Section* **??** *(an example of which is in* (**??**)*), what the next LIGR in the sequence could be by applying the backward search i.e.* F *as usual. This has to be done until closure i.e. until no more LIGR's can be found if this is repeated.*

While carrying this out, sequences of LIGR's are examined that give rise to the derivation of new LIGR's generated from them that can be next in the sequence. The set of possible sequences of LIGR's is defined by the set of LIGR's currently found and the rules determining what LIGR's can follow others. This is mainly but not entirely determined by the relation "can be followed by" or equivalently the relation "can be preceded by" on the set of LIGR's. Extra information can come from LIGR's that can prevent certain LIGR's to follow at two or more places further in the sequence. Thus the list of sequences of LIGR's has to be continually updated as the list of LIGR's is increased. This by both adding new sequences which consist of just one new LIGR, and adding a new preceding LIGR to any sequence.

In the results of this algorithm in (**??**) it would seem that many more sequences of LIGR's could be considered (in fact an infinite number) because the results give LIGR's that could be added on the right. This is not done because the updated set of sequences being considered is obtained as above

which requires new LIGR's that can precede previously known ones to be included. This implies that the sequences being considered are added to on the left, not the right. Anyway there is no point in using the previous idea because for many sequences, no preceding LIGR's can affect the new LIGR's that are produced by F so there is no point in considering them. This is indicated by the absence of RCS's, and limits the length of these sequences in the results.

In more detail, for any LIGR $\mathtt{X} \in \mathtt{L}$, find $\mathtt{F}_1(\mathtt{X})$ and if this is a new LIGR add it to the list, and only if F applied to X gives $\mathtt{F}_2(\mathtt{X}) \neq \emptyset$, carry out F again to obtain $\Delta\mathtt{F}_1(\mathtt{X}_1 \cdot \mathtt{X})$ and $\mathtt{F}_2(\mathtt{X}_1 \cdot \mathtt{X})$ for each LIGR $\mathtt{X}_1$ that can precede X where the requirement for precedence is given in (**??**). To obtain $\mathtt{F}_1$ and $\Delta\mathtt{F}_1$ the methods of subsection **??** apply if needed. This calculation can start from the previous $\mathtt{F}_2$ with the substitution made for T indicated by $\mathtt{X}_1$. Any new LIGR's from $\Delta\mathtt{F}_1(\mathtt{X}_1 \cdot \mathtt{X})$ are added to L. For any members of $\mathtt{F}_2(\mathtt{X}_1 \cdot \mathtt{X})$ apply this algorithm recursively i.e. with $\mathtt{X}_1 \cdot \mathtt{X}$ taking the place of X above etc.. It is expected that this algorithm will terminate because the matching criterion for new LIGR's that can be prepended to the sequence gets increasingly stringent as the length of the strings increases.

The condition $\mathtt{F}_2(\mathtt{X}) \neq \emptyset$ will be met for each of the initial set of LIGR's in L because as shown above their RHS's each have the form of an RCS.

It was convenient while doing this, for example TM1 with results in (**??**), to add new LIGR's to the end of the list, and use this ordering of LIGR's to order the sequences being analysed reverse lexicographically. This order is maintained when additional sequences are obtained.

The following are a few general results that are likely to be useful.

**Lemma 6.2.** *Suppose $\alpha$ is on the left and $\underline{\mathtt{T}_1}$ has the pointer at its leftmost symbol then:*
*If* F *applied to* $\mathtt{x}\underline{\mathtt{T}_1}$ *gives no RCS's then* F *applied to* $\mathtt{x}\underline{\mathtt{T}_1}\mathtt{T}_2$ *gives no RCS's.*
*The mirror image version with $\alpha$ on the right is:*
*If* F *applied to* $\mathtt{x}\underline{\mathtt{T}_1}$ *gives no RCS's then* F *applied to* $\mathtt{x}\mathtt{T}_2\underline{\mathtt{T}_1}$ *gives no RCS's.*

*Proof.* Start by assuming that applying F to $\mathtt{x}\underline{\mathtt{T}_1}\mathtt{T}_2$ with $\alpha$ on the left gives some RCS's, then there is a sequence of reverse TM steps starting from $\mathtt{x}\alpha\underline{\mathtt{T}_1}\mathtt{T}_2$ leading to a CS with the pointer at the right hand end. Then truncating the starting CS by $\mathtt{T}_2$ results in a truncated set of reverse TM steps leading to another CS which is the RCS for applying F to $\mathtt{x}\underline{\mathtt{T}_1}$ again with the pointer on the right. This contradicts the assumption therefore F applied to $\mathtt{x}\underline{\mathtt{T}_1}\mathtt{T}_2$ gives no RCS's. □

**Lemma 6.3.** *If applying* F *to* $\mathtt{x}\underline{\mathtt{T}_1}$ *generates the set of LIGR's* S *and no RCS's then applying* F *to* $\mathtt{x}\underline{\mathtt{T}_1}\mathtt{T}_2$ *generates likewise only the set* S *of LIGR's.*

**Lemma 6.4.** *If applying* F *to* $\mathtt{x}\alpha\underline{\mathtt{T}_1}$ *gives a set* $\mathtt{L}_1$ *of LIGR's then applying* F *to* $\mathtt{x}\alpha\underline{\mathtt{T}_1}\mathtt{T}_2$ *gives a set* $\mathtt{L}_2$ *of LIGR's such that* $\mathtt{L}_2 \supseteq \mathtt{L}_1$. *The mirror image*

*version is:*
*If applying* $\mathtt{F}$ *to* $\mathtt{x}\underline{\mathtt{T_1}}\alpha$ *gives a set* $\mathtt{L_1}$ *of LIGR's then applying* $\mathtt{F}$ *to* $\mathtt{x}\mathtt{T_2}\underline{\mathtt{T_1}}\alpha$ *gives a set* $\mathtt{L_2}$ *of LIGR's such that* $\mathtt{L_2} \supseteq \mathtt{L_1}$.

*Proof.* Applying $\mathtt{F}$ to $\mathtt{x}\underline{\mathtt{T_1}}$ gives a set of LIGR's determined by a set of reverse TM steps taking the pointer from where it starts in $\mathtt{x}\alpha\underline{\mathtt{T_1}}$ to a CS with the pointer at $\alpha$. By replacing the string $\mathtt{T_1}$ by $\mathtt{T_1}\mathtt{T_2}$ these reverse TM steps can still be carried out with the string $\mathtt{T_2}$ playing no role, but in addition there could be more such sequences of reverse TM steps giving rise to more LIGR's. □

Note that this result can sometimes be used to identify LIGR's produced by the program "origins" [**?**] using the values of $j_1$ produced.

**Lemma 6.5.** *If* $\mathtt{F}$ *applied to* $\mathtt{x}\underline{\mathtt{T_1}}$ *gives has an LIGR with the parameter value* $j_1$ *as defined in* (**??**)*, then* $\mathtt{F}$ *applied* $\mathtt{x}\underline{\mathtt{T_1}}$ *truncated to a string of length* $j_1 + \mathtt{1}$ *excluding* $\alpha$ *has some RCS's and is the longest such truncation guaranteed to have some RCS's because the pointer just reaches symbol* $\mathtt{y}'_{j_1+\mathtt{1}}$*. The truncation is such that it removes the symbols furthest from* $\alpha$*.*

This relates the largest value of $j_1$ given by the program "origins" [**?**] to the length of the string analysed (excludes $\alpha$).

**Definition 6.6.** *A finite set of LIGR's* $\mathtt{Z}$ *is closed under* $\mathtt{F}$ *if for any sequence of members of* $\mathtt{Z}$ *that can be substituted into one another in sequence as in* (**??**)*, the backward search* $\mathtt{F}$ *applied to the result of this generates results* $\mathtt{F_1}$ *that are each a member of the set* $\mathtt{Z}$ *and* $\mathtt{F_2} = \emptyset$*.*

**Theorem 6.7.** *If there is a finite set* $\mathtt{Z}$ *of LIGR's for a Turing Machine that is closed under* $\mathtt{F}$ *as described above and includes all the LIGR's involved in obtaining the set IRR(2) from the single TM steps, then every origin of an IRR for that TM can be obtained from the single TM steps by a sequence of applications of LIGR's each in* $\mathtt{Z}$ *as described in Section* **??***. This is obvious by considering the derivation of members of IRR*$(n + \mathtt{1})$ *from IRR(n) for all* $n \geq 2$*.*

It is also obvious that no LIGR's could be removed from this set $\mathtt{Z}$ and $\mathtt{Z}$ still have this property because members of $\mathtt{Z}$ are only put there when they are required.

This all assumes that $\mathtt{Z}$ is finite. The case if the closure algorithm does not terminate leading to an infinite set $\mathtt{Z}$ closed under $\mathtt{F}$ might be interesting but is not considered here.

That there is a finite number of LIGR's has been shown in the present case by the hand calculations summarised in Table **??**. Computer results (which rapidly increase in number and time taken as $n$ increases) established Table **??** using any value of the maximum length of the strings involved ($n$) between 10 and 16 and show the same result.

The relation "can be preceded by" on the LIGR's was being discovered as the LIGR's were being discovered. The criterion is that the RHS of the preceding LIGR must match the LHS of the original LIGR in state, string of symbols and direction. The relation "can be preceded by" among the initial LIGR's that arise from the derivation of the IRR(2) from the single TM steps, and these LIGR's themselves are given in (**??**) but note that some of these LIGR's appear in the final set ((**??**)) with the numbering slightly changed due to the use of `d` meaning `a` or `b` and other LIGR's including those of length 1 being found.

| ID | LIGR | can be preceded by |
|---|---|---|
| 1 | $2\underline{\mathtt{T}} \overset{\mathtt{b}}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{T}$ | 4, 9 |
| 2 | $3\underline{\mathtt{T}} \overset{\mathtt{b}}{\leftarrow} 1\mathtt{T}\underline{\mathtt{b}}$ | 6, 7 |
| 3 | $1\underline{\mathtt{T}} \overset{\mathtt{b}}{\leftarrow} 1\underline{\mathtt{c}}\mathtt{T}$ | 1, 3 |
| 4 | $2\underline{\mathtt{T}} \overset{\mathtt{c}}{\leftarrow} 2\underline{\mathtt{b}}\mathtt{T}$ | 4, 9 |
| 5 | $1\underline{\mathtt{T}} \overset{\mathtt{c}}{\leftarrow} 2\mathtt{T}\underline{\mathtt{c}}$ | 2 |
| 6 | $1\underline{\mathtt{T}} \overset{\mathtt{a}}{\leftarrow} 3\mathtt{T}\underline{\mathtt{a}}$ | 2 |
| 7 | $1\underline{\mathtt{T}} \overset{\mathtt{a}}{\leftarrow} 3\mathtt{T}\underline{\mathtt{b}}$ | 2 |
| 8 | $3\underline{\mathtt{T}} \overset{\mathtt{c}}{\leftarrow} 3\underline{\mathtt{c}}\mathtt{T}$ | 8 |
| 9 | $3\underline{\mathtt{T}} \overset{\mathtt{b}}{\leftarrow} 2\underline{\mathtt{a}}\mathtt{T}$ | 8 |

. (51)

## 6.1 A few useful lemmas etc.

While carrying out the main argument (the application of Algorithm **??** to the example (**??**)) in Section **??** the following results (lemmas) emerged and are collected here to avoid breaking up the logic of the grand search i.e. "depth first" in which they arose. They may need to be referred to anywhere throughout the main argument and speed up the computation of the results in this section. They arose as induction hypotheses to deal with endlessly repeating situations that arose or other complex situations arising there and so simplify the presentation of this. In all these `x` is a state and $\mathtt{T}_1$ and $\mathtt{T}_2$ are strings.

It will be useful to use the symbol `d` in the remainder of this paper to mean either `a` or `b`. This arose as an abbreviation useful for TM (**??**). Each instance of `d` will be independent of any other one in a CS, so that all combinations are possible. If the combinations that are possible are a subset of these, this is more complicated and alternatives will be given in braces, but this does not usually happen. This will allow many cases to be considered simultaneously so speeding up the computations and shortening the notation. When doing reverse computations, all possible reverse steps from any of the combinations will be included.

**Lemma 6.8.** *The reversed TM cannot cross where the symbol* `c` *is while going right.*

*Proof.* If the pointer is just left of the symbol c i.e. in the CS _c a reverse step to the right is only possible if it is to the CS 2c (using 2c → 1_c in reverse). The next reverse TM step (if one is possible) must be to the left because there are no TM steps of the form xz → 2_y for any state x and symbol z that would take the reversed TM to the right. Thus the symbol c is maintained and the pointer has not crossed it. □

**Lemma 6.9.** *The reversed TM cannot cross where the symbol* a *is while going left, or even arrive at it.*

*Proof.* There is no reverse TM step of the form xa_ ← CS therefore the pointer cannot get left of or arrive at where the a is, which is maintained. □

**Corollary 6.10.** *A consequence of this is that an* a *in the CS string to the left of the pointer restricts the search for LIGR's and RCS's to the substring to the right of the* a*. Therefore if while doing the backward search a CS arises which is actually a set of CS's that looks like a single CS's because it contains symbols* d *that can be* a *or* b*, it is possible treat this as a set of cases starting with the* d *to the immediate left of the pointer being* a*. In each case one more of the* d*'s is replaced by* b*, and the* d *to the immediate left of this* d *(if it exists) is* a *and proceeds going left by one* d *at a time. The first case has the simplest analysis under* F*, and ends with the case where all the* d*'s are* b*. For the case when there are four* d*'s the following cases arise in this order*

$$\begin{array}{l} \texttt{d}\cdots\texttt{d}\cdots\texttt{d}\cdots\texttt{a} \\ \texttt{d}\cdots\texttt{d}\cdots\texttt{a}\cdots\texttt{b} \\ \texttt{d}\cdots\texttt{a}\cdots\texttt{b}\cdots\texttt{b} \\ \texttt{a}\cdots\texttt{b}\cdots\texttt{b}\cdots\texttt{b} \\ \texttt{b}\cdots\texttt{b}\cdots\texttt{b}\cdots\texttt{b} \end{array} .$$

*Lemma* **??** *will simplify each of these analyses in fact if* $\alpha$ *is on the right, no RCS's can result except for the last case, and the sets of LIGR's produced are nested such that they are all produced by the analysis of the last case alone.*

See the examples in Section **??**

### 6.1.1 Using the values of $j_1$

Because of the fact that LIGR's result from backward searches where the pointer starts and ends up at $\alpha$, any such result can be usefully classified by the parameter $j_1$ in (**??**), the maximum number of reverse TM steps away from $\alpha$ during the computation. The resulting LIGR will have length (on the right) equal to $j_1 + 2$ because the symbol where the computation starts and $\alpha$ have to be included. Therefore in the output of the program [**?**], the LIGR's can often be identified by just using the values of $j_1$.

## 6.2 The main argument

This long section contains the details of the application of Algorithm **??** to TM (**??**) which is summarised in Table **??** and again in Figure **??**. The problem

with presenting this arose from the decision to present this in reverse lexicographical order of the CS analysed instead of logical order of derivation. This results in a much neater arrangement but it contains forward references. In the following, the ID numbers of LIGR's refer to the LIGR's listed in (**??**) which is the most up to date list of LIGR's obtained, for which the relation "can be preceded by" is given in (**??**).

Rather than repeating the phrase "Applying F to the sequence of LIGR's X gives ..." on many occasions it will be shortened to $\mathtt{X} \overset{\mathtt{F}}{\Rightarrow} \ldots$.

### 6.2.1 Sequences ending with LIGR 1

Applying F to 1 gives just $1\alpha\underline{a}T \overset{b}{\leftarrow} 1\underline{c}aT$ i.e. LIGR 3 i.e. $1 \overset{F}{\Rightarrow} 3$. Clearly applying F to an LIGR of length one as is done here could possibly lead to more results if T is specialised by giving a symbol at one end of the string (here the left end) because the reversed TM might then take a step to the right which cannot be determined until the leftmost symbol of T is known. Therefore preceding LIGR's must be considered. LIGR 1 can be preceded by LIGR 4 giving $4 \cdot 1$ having the combined effect $3\underline{T} \overset{b}{\leftarrow} 2\underline{a}T \overset{b}{\leftarrow} 1\underline{a}aT$ i.e. $3\underline{T} \overset{bb}{\leftarrow} 1\underline{a}aT$ and $4 \cdot 1 \overset{F}{\Rightarrow} 3$ because F gives the calculation

$$1\alpha\underline{a}aT \begin{cases} \overset{b}{\leftarrow} 1\underline{c}aaT \\ \leftarrow \begin{cases} 3\alpha a\underline{a}T \\ 3\alpha a\underline{b}T \end{cases} \end{cases} \text{ i.e. } 1\alpha\underline{a}aT \begin{cases} \overset{b}{\leftarrow} 1\underline{c}aaT \\ \leftarrow 3\alpha a\underline{d}T \end{cases} \tag{52}$$

The first part of this is LIGR 3 as found before with aT taking the place of T, and the second part is a pair of RCS's. By specialising this further by giving any more symbols of T, the first result will obviously not generate any new LIGR's after reducing the result to its shortest form, the result will merely be replicated. However for the second part it is possible that the reverse computation could take the pointer back to $\alpha$ by specifying the next symbol of T and so generate more new LIGR's, therefore the search has to continue back, so we have thus far one RCS and

$$\begin{matrix} \ldots 1 \overset{F}{\Rightarrow} \{3\} \\ \ldots \cdot 4 \cdot 1 \overset{F}{\Rightarrow} \{3\} \end{matrix}. \tag{53}$$

Because there are RCS's, any LIGR's that can precede $4 \cdot 1$ must be considered and F must be applied to all these. The LIGR's that can precede 4 are just 8,10 and 21. The sequence $8 \cdot 4 \cdot 1$ gives $3\underline{T} \overset{c}{\leftarrow} 3\underline{c}T \leftarrow 1\underline{a}acT$. Applying F to this can start from (**??**) with the substitution given by LIGR $8\ 1\alpha\underline{a}acT \leftarrow 3\alpha a\underline{d}cT$ in addition to 3 as above and $\Delta F_1$ is just the result of this backward search from $3\alpha a\underline{d}cT$ and because the computation cannot go back from there $\Delta F_1(8, 4 \cdot 1) = \emptyset$ and $F_2(8 \cdot 4 \cdot 1) = \emptyset$ using the definitions in Section **??**. Therefore there are no additional results of F (LIGR's or RCS's)

and it is now it is clear that no preceding LIGR's specialising this $\mathtt{T}$ can give any such new results so the search for new LIGR's stops in this branch of the grand search tree which has to continue with $\mathtt{10\cdot 4\cdot 1}$. This sequence has the effect $\mathtt{1\underline{c}aT} \leftarrow \mathtt{1\underline{a}accdT}$ and applying $\mathtt{F}$ gives $\mathtt{1}\alpha\mathtt{\underline{a}accdT} \leftarrow \mathtt{3}\alpha\mathtt{\underline{a}\underline{d}ccdT}$ (apart from the above results from $\mathtt{4\cdot 1}$) from which there are no further backward steps so there are no new LIGR's or RCS's and these branches of the grand search end i.e. $\Delta\mathtt{F}_1(\mathtt{10\cdot 4\cdot 1}) = \mathtt{F}_2(\mathtt{10\cdot 4\cdot 1}) = \emptyset$. $\mathtt{21\cdot 4\cdot 1}$ has the effect $\mathtt{1\underline{c}abcT} \leftarrow \mathtt{1\underline{a}ac^3dcT}$ and $\mathtt{F}$ applied to this by (**??**) (apart from the above results from $\mathtt{4\cdot 1}$) gives $\mathtt{1}\alpha\mathtt{\underline{a}ac^3dcT} \leftarrow \begin{cases}\mathtt{3}\alpha\mathtt{a\underline{a}c^3dcT}\\ \mathtt{3}\alpha\mathtt{a\underline{b}c^3dcT}\end{cases}$ which by Lemma **??** cannot lead to an LIGR. Note that this is first of many examples of the use of Lemma **??** that was not anticipated in the general algorithm **??** and was found as a result of its systematic use combined with spotting a common situation namely the reversed TM having to cross the symbol $\mathtt{c}$ while going right which is impossible. This completes the analysis for all sequences that can precede $\mathtt{4\cdot 1}$ therefore the next sequence of LIGR's to be considered in the grand search is $\mathtt{5\cdot 1}$ i.e. $\mathtt{2\underline{T}} \leftarrow \mathtt{2\underline{b}T} \leftarrow \mathtt{1\underline{a}bT}$. The computation of $\mathtt{F}$ starts from $\mathtt{1}\alpha\mathtt{\underline{a}bT}$ and gives no result other than LIGR $\mathtt{3}$, so $\Delta\mathtt{F}_1(\mathtt{5},\mathtt{1}) = \mathtt{F}_2(\mathtt{5\cdot 1}) = \emptyset$. Next $\mathtt{9\cdot 1}$ must be considered which is $\mathtt{1\underline{c}aT} \leftarrow \mathtt{2\underline{a}cdT} \leftarrow \mathtt{1\underline{a}acdT}$.

$$\mathtt{9\cdot 1} \stackrel{\mathtt{F}}{\Rightarrow} \mathtt{1}\alpha\mathtt{\underline{a}acdT} \leftarrow \begin{cases}\mathtt{1\underline{c}aacdT}\\ \mathtt{3}\alpha\mathtt{a\underline{d}cdT}\end{cases}. \tag{54}$$

which is also a special case of (**??**). The first of these is just LIGR $\mathtt{3}$ and the second cannot continue, so there are no new LIGR's from preceding $\mathtt{1}$ by $\mathtt{9}$ i.e. $\Delta\mathtt{F}_1(\mathtt{9},\mathtt{1}) = \mathtt{F}_2(\mathtt{9\cdot 1}) = \emptyset$.

Using Lemma **??** and Lemma **??** it is easy to show that similar results hold for all the other LIGR's that could precede LIGR $\mathtt{1}$ and the results can be summarised as

$$\Delta\mathtt{F}_1(\{\mathtt{5},\mathtt{9},\mathtt{12},\mathtt{14},\mathtt{20},\mathtt{29},\mathtt{31}\},\mathtt{1}) = \mathtt{F}_2(\{\mathtt{5},\mathtt{9},\mathtt{12},\mathtt{14},\mathtt{20},\mathtt{29},\mathtt{31}\}\cdot\mathtt{1}) = \emptyset. \tag{55}$$

This exhausts all search trees in the grand search starting from LIGR $\mathtt{1}$.

### 6.2.2 Sequences ending with LIGR 2

Next consider applying $\mathtt{F}$ to sequences ending with $\mathtt{2}$ which is $\mathtt{3\underline{T}} \stackrel{\mathtt{b}}{\leftarrow} \mathtt{1T\underline{b}}$. $\mathtt{F}$ applied to this gives

$$\mathtt{1T\underline{b}}\alpha\begin{cases}\stackrel{\mathtt{a}}{\leftarrow} \mathtt{3Tb\underline{d}}\\ \stackrel{\mathtt{c}}{\leftarrow} \mathtt{2Tb\underline{c}}\end{cases} \tag{56}$$

which are LIGR's $\mathtt{6}$ and $\mathtt{7}$ so

$$\mathtt{2} \stackrel{\mathtt{F}}{\Rightarrow} \{\mathtt{6},\mathtt{7}\}. \tag{57}$$

The last symbol of $\mathtt{T}$ in (**??**) could affect this result so LIGR's preceding $\mathtt{2}$ must be considered which are just $\mathtt{7}$,$\mathtt{16}$,$\mathtt{18}$,$\mathtt{22}$,$\mathtt{24}$ and $\mathtt{26}$. Consider $\mathtt{7\cdot 2}$ which

is $1\underline{T} \overset{a}{\leftarrow} 3T\underline{d} \leftarrow 1Td\underline{b}$. F gives $1Td\underline{b}\alpha \leftarrow 1T\underline{c}b\alpha$ in addition to 6 and 7 and so can be potentially specialised further i.e. $\Delta F_1(7,2) = \emptyset$ and $F_2(7 \cdot 2) = 1T\underline{c}b\alpha$. LIGR 7 can only be preceded by 2 and 15 so the next sequence to be considered in the grand search is $2 \cdot 7 \cdot 2$ which is $3\underline{T} \overset{b}{\leftarrow} 1T\underline{b} \leftarrow 1Td\underline{b}$. F applied to this gives

$$1Td\underline{b}\alpha \leftarrow 1Tb\underline{c}b\alpha \leftarrow 1T\underline{c}cb\alpha \tag{58}$$

i.e. $\Delta F_1(2, 7 \cdot 2) = \emptyset$ (using Lemma **??**) and $F_2(2 \cdot 7 \cdot 2) = 1T\underline{c}cb\alpha$. This RCS by Lemma (**??**) cannot lead to any new LIGR's because the pointer can never reach $\alpha$ by the reverse TM computation however many preceding LIGR's are added to the sequence. Therefore there is no point in continuing grand search along this branch. Next consider $15 \cdot 7 \cdot 2$ having the effect $3Tb^5\underline{a} \leftarrow 1Tc\begin{Bmatrix} db \\ aa \end{Bmatrix}dbdbd\underline{b}$. Applying F to this gives a result of the same form as in (**??**) therefore the same conclusion follows and the grand search continues from $16 \cdot 2$ which has the effect $3Tb^5\underline{a} \leftarrow 1Tca^3dbd\underline{b}$. Applying F to this again gives results of the form (**??**) not leading to any new results for LIGR's.

The sequence $18{\cdot}2$ has the effect $3Tb^5\underline{a} \leftarrow 3Tcd\begin{Bmatrix} ba \\ cb \end{Bmatrix}db\underline{d} \leftarrow 1Tcd\begin{Bmatrix} ba \\ cb \end{Bmatrix}dbd\underline{b}$. Applying F to this gives $1Tcd\begin{Bmatrix} ba \\ cb \end{Bmatrix}dbd\underline{b}\alpha \overset{b}{\leftarrow} 1Tcd\begin{Bmatrix} ba \\ cb \end{Bmatrix}db\underline{c}b\alpha \leftarrow 1Tcd\begin{Bmatrix} ba \\ cb \end{Bmatrix}d\underline{c}cb\alpha$. There is no point going any further with the algorithm F because from this CS, by Lemma (**??**) it is not possible for the pointer to get to $\alpha$ regardless of any preceding LIGR's i.e. no new LIGR's can result from this, another unanticipated short cut to Algorithm **??**. Similar results all hold for $\{22, 24, 26\} \cdot 2$ which all lead to applying the backward search from a CS of the form $1Tdbd\underline{b}\alpha$.

### 6.2.3 Sequences ending with LIGR 3

Consider sequences of LIGR's ending with 3 which is $1\underline{T} \overset{b}{\leftarrow} 1\underline{c}T$. Applying F starts from $1\alpha\underline{c}T \overset{b}{\leftarrow} 1\underline{c}cT$ showing that $3 \overset{F}{\Rightarrow} \{3\}$. The LIGR 3 can be preceded by 1, 3, 11, 13, 28 and 30. The sequence $1{\cdot}3$ is $2\underline{T} \leftarrow 1\underline{a}T \leftarrow 1\underline{c}aT$. Applying F starts from $1\alpha\underline{c}aT\begin{cases} \overset{b}{\leftarrow} 1\underline{c}caT \\ \leftarrow 3\alpha c\underline{d}T \end{cases}$ showing that two new RCS's are the only extra results of preceding 3 with 1. The sequence $4 \cdot 1 \cdot 3$ is $3\underline{T} \leftarrow 2\underline{a}T \leftarrow 1\underline{c}aaT$. Applying F gives

$$1\alpha\underline{c}aaT \leftarrow 3\alpha c\underline{d}aT \leftarrow \ldots \begin{cases} \leftarrow 3\alpha cb\underline{d}T \\ \overset{b}{\leftarrow} 2\underline{a}cdaT\,. \\ \overset{c}{\leftarrow} 3\underline{c}cdaT \end{cases} \tag{59}$$

where the pointer does not reach the a adjacent to T during this computation of the last two parts, therefore these shorten to

$$1\underline{c}aT\begin{cases}\overset{b}{\leftarrow} 2\underline{a}cdT\\ \overset{c}{\leftarrow} 3\underline{c}cdT\end{cases} \tag{60}$$

which are new LIGR's and will be numbered 9 and 10 respectively, and the RCS's which require further backward searching. The sequence $8\cdot 4\cdot 1\cdot 3$ is $3\underline{T}\leftarrow 3\underline{c}T\leftarrow 1\underline{c}aacT$. Applying F gives

$$1\alpha\underline{c}aacT\leftarrow 3\alpha cb\underline{d}cT\begin{cases}\overset{b}{\leftarrow} 1\underline{a}badcT\\ \overset{c}{\leftarrow} 2\underline{b}badcT\end{cases} \tag{61}$$

after a few steps. These when abbreviated are

$$1\underline{c}aaT\begin{cases}\overset{b}{\leftarrow} 1\underline{a}badT\\ \overset{c}{\leftarrow} 2\underline{b}badT\end{cases} \tag{62}$$

which will be numbered LIGR's 11 and 12 respectively. Also there are now no RCS's, so this branch of the grand search ends.

The sequence $10\cdot 4\cdot 1\cdot 3$ has the effect $1\underline{c}aT\leftarrow 1\underline{c}aaccdT$ and F gives, using (**??**).1,

$$1\alpha\underline{c}aaccdT\leftarrow 3\alpha cb\underline{d}ccdT\begin{cases}\overset{b}{\leftarrow} 1\underline{a}badccdT\\ \overset{c}{\leftarrow} 2\underline{b}badccdT\end{cases} \tag{63}$$

and produces only 11 and 12 again and no new RCS's.

The analysis for $21\cdot 4\cdot 1\cdot 3$ is similar with the same result. Next is $5\cdot 1\cdot 3$ with the effect $2\underline{T}\leftarrow 1\underline{c}abT$. Then applying F gives

$$1\alpha\underline{c}abT\leftarrow 3\alpha c\underline{d}bT\leftarrow\begin{cases}3\alpha\underline{c}dbT\begin{cases}\overset{b}{\leftarrow} 2\underline{a}cdbT\\ \overset{c}{\leftarrow} 3\underline{c}cdbT\end{cases}\\ 1\alpha cd\underline{b}T\end{cases} \tag{64}$$

apart. This gives two results which reduce to LIGR's 9 and 10 again and an RCS. Next is $4\cdot 5\cdot 1\cdot 3$ which is $3\underline{T}\leftarrow 1\underline{c}abaT$ and F gives

$$1\alpha\underline{c}abaT\leftarrow 1\alpha cd\underline{b}aT\leftarrow 3\alpha cdb\underline{d}T \tag{65}$$

using (**??**) apart from one branch giving a CS which cannot be continued back. This RCS requires going back in the grand search.

The next sequence is $8\cdot 4\cdot 5\cdot 1\cdot 3$ giving $3\underline{T}\leftarrow 1\underline{c}abacT$. Because of Lemma **??** backward searching from here cannot lead to any RCS's and F gives (using (**??**))

$$1\alpha\underline{c}abacT\leftarrow 3\alpha cdb\underline{d}cT\overset{3}{\leftarrow} 3\alpha ca\underline{d}dcT\leftarrow 2\alpha cadb\underline{c}T. \tag{66}$$

By Lemma **??** because of the a on the left of the pointer, no new LIGR's can result if this branch of the grand search is continued and because of the c on the right no RCS's result. From the next sequence $10 \cdot 4 \cdot 5 \cdot 1 \cdot 3$ which has the effect $1\underline{c}aT \leftarrow 1\underline{c}abaccdT$, F can start (by (**??**)) from $2\alpha cadb\underline{c}cdT$ and again by Lemma **??** no LIGR's or RCS's can result from this branch. The same holds for $21 \cdot 4 \cdot 5 \cdot 1 \cdot 3$ because the results are special cases of (**??**). The next sequence is $5 \cdot 5 \cdot 1 \cdot 3$, and F can start from $1\alpha\underline{c}abbT$ which is a special case of (**??**) and picking out the result that does not lead to an LIGR already found gives

$$1\alpha\underline{c}abbT \leftarrow 1\alpha cd\underline{b}bT \leftarrow 1\alpha c\underline{c}bbT \tag{67}$$

from which there are no more reverse TM steps are possible so there are no RCS's or LIGR's.

Applying F to $9 \cdot 5 \cdot 1 \cdot 3$ can start from $1\alpha\underline{c}abacdT$ i.e. (**??**) with dT in the place of T so this gives $2\alpha cadb\underline{c}T \leftarrow 1\alpha cad\underline{a}cdT$ from which there are no RCS's or new LIGR's.

$12 \cdot 5 \cdot 1 \cdot 3$ is $1\underline{c}aaT \leftarrow 2\underline{b}badT \leftarrow 1\underline{c}abbbadT$ for which the analysis with F is a special case of (**??**) so there are no RCS's or LIGR's produced by F.

$14 \cdot 5 \cdot 1 \cdot 3$ is $1\underline{c}cT \leftarrow 2\underline{b}bcT \leftarrow 1\underline{c}ab^3cT$. The same argument holds here. For $20 \cdot 5 \cdot 1 \cdot 3$ which its application of F can start from $1\alpha\underline{c}abaccdcT \leftarrow 2\alpha cadb\underline{c}cdcT$ using (**??**), which by Lemma **??** cannot generate any more LIGR's. Both of $\{29, 31\} \cdot 5 \cdot 1 \cdot 3$ lead to special cases of (**??**) giving no more LIGR's or RCS's. $12 \cdot 1 \cdot 3$ is $1\underline{c}aaT \leftarrow 2\underline{b}badT \leftarrow 1\underline{c}abbadT$. This again is a special case of (**??**) showing that there are no more results. Likewise for each of $\{14, 29, 31\} \cdot 1 \cdot 3$. For $20 \cdot 1 \cdot 3$ its application of F starts from $1\alpha\underline{c}aaccdcT$ which is a special case of (**??**) and gives no new results.

The sequence $3 \cdot 3$ has the effect $1\underline{T} \overset{b}{\leftarrow} 1\underline{c}T \leftarrow 1\underline{c}cT$. F gives $1\alpha\underline{c}cT \leftarrow 2\alpha c\underline{c}T$ an RCS so continue. $1 \cdot 3 \cdot 3$ has the effect $2\underline{T} \overset{b}{\leftarrow} 1\underline{a}T \leftarrow 1\underline{c}caT$ and F gives

$$1\alpha\underline{c}caT \leftarrow 2\alpha c\underline{c}aT \leftarrow 2\alpha\underline{b}caT \begin{cases} \overset{b}{\leftarrow} 1\underline{a}bcaT \\ \overset{c}{\leftarrow} 2\underline{b}bcaT \end{cases} \tag{68}$$

which shortens to

$$1\underline{c}cT \leftarrow \begin{cases} 1\underline{a}bcT \\ 2\underline{b}bcT \end{cases} \tag{69}$$

(because the rightmost position of the pointer in this derivation is just before the substring aT that is unaltered) and gives no RCS's. Equation (**??**) will be called LIGR's 13 and 14 respectively. Consider the next sequence $3 \cdot 3 \cdot 3$ with effect $1\underline{T} \overset{b}{\leftarrow} 1\underline{c}T \leftarrow 1\underline{c}ccT$ which is a special case of (**??**). Next are $\{11, 13, 28, 30\} \cdot 3 \cdot 3$ which due to Lemma **??** only give LIGR 3 with F. Next consider $11 \cdot 3$ which is $1\underline{c}aaT \leftarrow 1\underline{c}abadT$ which, $\leftarrow 3\alpha cdb\underline{d}dT$ by (**??**) which, continuing with F in 3 steps gives $3\alpha ca\underline{d}ddT$, which by Lemma **??** gives no more LIGR's, and in one step gives the RCS $1\alpha cdbd\underline{b}T$. Continuing, the next sequence is $3 \cdot 11 \cdot 3$ which is $1\underline{a}aT \leftarrow 1\underline{c}abadT$ so applying F is as above.

Continuing, $1\cdot 3\cdot 11\cdot 3$ is $2\underline{a}T \leftarrow 1\underline{c}abadT$ again giving the same result. Again, $4\cdot 1\cdot 3\cdot 11\cdot 3$ gives $3\underline{T} \leftarrow 1\underline{c}abadT$ and the same result with F. $\{8,10,21\}\cdot 4\cdot 1\cdot 3\cdot 11\cdot 3$ gives $3\underline{T} \leftarrow 1\underline{c}abadcT'$ so by Lemma **??** so no new LIGR's from F can be generated on these branches. For the sequence $5\cdot 1\cdot 3\cdot 11\cdot 3$ the LIGR's don't match because 5 is $2\underline{T} \leftarrow 2\underline{b}T$ whereas $1\cdot 3\cdot 11\cdot 3$ starts from $2\underline{a}T$. Likewise $\{12,14,29,31\}\cdot 1\cdot 3\cdot 11\cdot 3$ are excluded. Both $\{9,20\}\cdot 1\cdot 3\cdot 11\cdot 3$ give results that are the same by Lemma **??** because of the c's in position 6 and the sequences are the same up to that point. A long calculation with up to 13 TM steps is needed to show that the LIGR's produced are $3,9,10,28-31$ with no RCS's. The sequence $3\cdot 11\cdot 3$ cannot be preceded by any of $3,11,13,28,30$ so the next case to be considered is $13\cdot 3$ which is $1\underline{c}cT \leftarrow 1\underline{a}bcT \leftarrow 1\underline{c}abcT$. F gives (apart from the first reverse step to the left giving LIGR 3)

$$1\alpha\underline{c}abcT \leftarrow \begin{cases} 2\underline{a}cdbcT \\ 3\underline{c}cdbcT \\ 2\alpha cdb\underline{c}T \end{cases} \tag{70}$$

which when shortened give LIGR's 9 and 10 and an RCS. Only 3 can precede 13 and $3\cdot 13\cdot 3$ is $1\underline{c}T \leftarrow 1\underline{c}cT \leftarrow 1\underline{c}abcT$. Note that here an extra symbol c was needed in order that the RHS of 3 matched the LHS of $13\cdot 3$. F gives the same result as above. Of the LIGR's that can precede 3 only 3 is compatible because of the symbol c on left and this gives $3\cdot 3\cdot 13\cdot 3$ which is $1\underline{T} \leftarrow 1\underline{c}T \leftarrow 1\underline{c}abcT$ which again gives the same result of F. $1\cdot 3\cdot 3\cdot 13\cdot 3$ is $2\underline{T} \leftarrow 1\underline{a}T \leftarrow 1\underline{c}abcaT$. Applying F (using (**??**)) can start from $2\alpha cdb\underline{c}aT$ (apart from LIGR's already found) and gives no RCS's and LIGR's 20 and 21. $3\cdot 3\cdot 3\cdot 13\cdot 3$ is $1\underline{T} \leftarrow 1\underline{c}T \leftarrow 1\underline{c}abccT$. Likewise F gives LIGR's 20 and 21 and no RCS's because these computations are the same because the pointer never gets to the symbol just left of T, and likewise for $11\cdot 3\cdot 3\cdot 13\cdot 3$ which is $1\underline{c}aaT \leftarrow 1\underline{c}abcabadT$ and $13\cdot 3\cdot 3\cdot 13\cdot 3$ which is $1\underline{c}cT \leftarrow 1\underline{c}abcabcT$ and $\{28,30\}\cdot 3\cdot 3\cdot 13\cdot 3$ by Lemma **??**. The sequences $\{11,13,28,30\}\cdot 3\cdot 13\cdot 3$ are not possible because of other compatibility restrictions based non-adjacent LIGR's. The sequences $28\cdot 3$ and $30\cdot 3$ both give results that are special cases of (**??**) giving no RCS's or LIGR's.

### 6.2.4 Sequences ending with LIGR 4

LIGR 4 is $3\underline{T} \overset{b}{\leftarrow} 2\underline{a}T$. F gives

$$2\alpha\underline{a}T\begin{cases} \overset{b}{\leftarrow} 1\underline{a}aT \\ \overset{c}{\leftarrow} 2\underline{b}aT \end{cases} \tag{71}$$

which shortens to 1 and 5, and this result could depend on the leftmost symbol of T because right-moving reverse steps could occur. Therefore LIGR's preceding 4 must be considered. $8\cdot 4$ is $3\underline{T} \leftarrow 3\underline{c}T \leftarrow 2\underline{a}cT$, and applying

F gives no new LIGR's (i.e. excluding 1 and 5 above) or RCS's. $10 \cdot 4$ is $1\underline{c}aT \leftarrow 3\underline{c}cdT \leftarrow 2\underline{a}ccdT$ and F gives no new LIGR's or RCS's and likewise for $21 \cdot 4$ which is $1\underline{c}abcT \leftarrow 2\underline{a}cccdcT$ because of Lemma **??**. Therefore these results show that $\{8, 10, 21\} \cdot 4 \overset{F}{\Rightarrow} \{1, 5\}$ only.

### 6.2.5 Sequences ending with LIGR 5

LIGR 5 is $2\underline{T} \leftarrow 2\underline{b}T$ and applying F gives $2\alpha\underline{b}T \begin{cases} \overset{b}{\leftarrow} 1\underline{a}bT \\ \overset{c}{\leftarrow} 2\underline{b}bT \end{cases}$ which shortens to 1 and 5 so $5 \overset{F}{\Rightarrow} \{1, 5\}$. The sequence $4 \cdot 5$ is $3\underline{T} \leftarrow 2\underline{a}T \leftarrow 2\underline{b}aT$. Applying F starts from $2\alpha\underline{b}aT$ and gives no new LIGR's and no RCS's. This is likewise true for 5,9,12,14,20,29 and 31 preceding 5 and all follow from the fact that 2_a and 2_b cannot be arrived at from a step of the TM, so all sequences ending with $\{4, 5, 9, 12, 14, 20, 29, 31\} \cdot 5 \overset{F}{\Rightarrow} \{1, 5\}$.

### 6.2.6 Sequences ending with LIGR 6

LIGR 6 is $1\underline{T} \leftarrow 2T\underline{c}$. Applying F gives $2T\underline{c}\alpha \leftarrow \emptyset$. A left-moving reverse step could occur depending on the rightmost symbol of T so this needs to be specialised by considering all possible previous LIGR's i.e 2 and 15. The sequence $2 \cdot 6$ is $3\underline{T} \leftarrow 1T\underline{b} \leftarrow 2Tb\underline{c}$ and applying F gives $2Tb\underline{c}\alpha \leftarrow 1T\underline{a}c\alpha$. By Lemma **??** this cannot lead to new LIGR's because $\alpha$ can never be reached by the pointer however the string is specialised by preceding LIGR's in the sequence. Likewise for $15 \cdot 6$.

Note that Lemma **??** was first discovered after studying this case in particular applying F to sequences of the form $2 \cdot (8 \cdot 2)^k \cdot 6$ that arose during the application of Algorithm **??** using an induction argument.

### 6.2.7 Sequences ending with LIGR 7

This starts with applying F to 7 i.e. $1\underline{T} \leftarrow 3T\underline{d}$ and gives $3T\underline{d}\alpha \leftarrow 1Td\underline{b}$ shortening to LIGR 2 because the d plays no role. Because the first LIGR preceding LIGR 2 is LIGR 7 and conversely, the same way of reasoning generates at first alternating sequences of 2 and 7 which can be analysed using Lemma **??** and its corollary which was derived with this example in mind. It also applies to other cases. Applying this to the sequence $15 \cdot 7$ with combined effect $3Tb^5\underline{a} \leftarrow 3Tc \begin{Bmatrix} db \\ ca \end{Bmatrix} dbdb\underline{d}$, if the last but one d is a the only possible reverse search paths for applying F to $15 \cdot 7$ are confined to the string $b\underline{d}$ restricting the set of LIGR's produced and no RCS's can be produced. If the last but one d is b and the last but two d is a, then the CS ends with $3Tabbb\underline{d}$ and the a cannot be reached in the backward search for F. Again no RCS's can be produced by F and the LIGR's produced are a superset of those produced

in the above case by Lemma **??**. Now suppose these d's are both b and the lower element of the array is taken, then the first a will again prevent any RCS's being produced and the LIGR's will be from 3Tabbbbd again a superset of the preceding case. Now taking the upper element in the array with the first d being a gives again no RCS's and the LIGR's are from 3Tabbbbbd again a superset of the preceding case. Finally, this leaves the case $\mathtt{3Tcb}^{6}\mathtt{d}\alpha$ which generates LIGR's which are again a superset of those from the preceding case.

Using this as starting point with the two cases where the d is a and b, F gives, using the program "origins" [**?**], a long list of results for case a. It gives 43 LIGR parts which exactly account for LIGR's $15 - 19$, and LIGR 2 which corresponds to the single result with $\mathtt{y.j1} = 1$. The 19 RCS's all have a c that must be crossed to get to $\alpha$ which is impossible because of Lemma **??**, therefore these RCS's can be ignored because they can never lead to more LIGR's. The case $\mathtt{3Tcb}^{6}\mathtt{b}$ needs to be considered too. Applying the program again and using Lemma **??** gives LIGR 2 and 24 other LIGR parts which correspond to LIGR's $22 - 27$, and no RCS's except those having a c that needs to be crossed that can be ignored.

A very similar argument works for applying F to any sequence of LIGR's that have an effect (column 2 of Table **??**) originating from a CS of the form $\mathtt{3T}(\mathtt{bd})^{3}\mathtt{d}$ showing that the same LIGR's are produced and no usable RCSs. This applies to the following sequences: $\{16, 18, 22, 24, 26\} \cdot (2 \cdot 7)^{3}, 15 \cdot (7 \cdot 2)^{2} \cdot 7, \{16, 18, 22, 24, 26\} \cdot 2 \cdot 7 \cdot 2 \cdot 7, 15 \cdot 7 \cdot 2 \cdot 7$.

Similar arguments also work for the remaining cases ending with LIGR 7 summarised in the following tables.

Table 4: LIGR's and RCS's

| LIGR sequence | CS analysed | RCS's produced | LIGR's produced |
|---|---|---|---|
| $16 \cdot 2 \cdot 7$ | $\leftarrow 3\mathtt{Tca}^3\mathtt{dbdb}\underline{\mathtt{d}}$ | $\emptyset$ | from $3\mathtt{Tab}^4\underline{\mathtt{d}} \subseteq$ from $3\mathtt{Tb}^6\underline{\mathtt{d}} =$ $\{2, 15-19, 22-27\}$ |
| $18 \cdot 2 \cdot 7$ | $\leftarrow 3\mathtt{Tcd}\left\{\begin{matrix}\mathtt{ba}\\ \mathtt{cb}\end{matrix}\right\}\mathtt{dbdb}\underline{\mathtt{d}}$ | none useful | from $3\mathtt{Tcdcb}^5\underline{\mathtt{d}} =$ $\{2, 15-19, 22-27\}$ (see Table ??) |
| $22 \cdot 2 \cdot 7$ | $\leftarrow 3\mathtt{Tadbdb}\underline{\mathtt{d}}$ | $\emptyset$ | from $3\mathtt{Tab}^4\underline{\mathtt{d}}$ |
| $24 \cdot 2 \cdot 7$ | $\leftarrow 3\mathtt{Tcbdbdb}\underline{\mathtt{d}}$ | none useful | from $3\mathtt{Tcb}^5\underline{\mathtt{d}} \subseteq$ from $3\mathtt{Tcdcb}^5\underline{\mathtt{d}}$ |
| $26 \cdot 2 \cdot 7$ | $\leftarrow 3\mathtt{Tcadbdbdb}\underline{\mathtt{d}}$ | $\emptyset$ | from $3\mathtt{Tab}^6\underline{\mathtt{d}} =$ from $3\mathtt{Tb}^6\underline{\mathtt{d}}$ |

Table 5: LIGR's generated by the computer program "origins" [**?**]

| $3\mathtt{Tcdcb}^5\underline{\mathtt{a}}$ | | | $3\mathtt{Tcdcb}^5\underline{\mathtt{b}}$ | | |
|---|---|---|---|---|---|
| $\mathtt{j}_1$ | # | LIGR's | $\mathtt{j}_1$ | # | LIGR's |
| 0 | 1 | 2 | 0 | 1 | 2 |
| 5 | 42 | $15-19$ | 2 | 6 | $22, 23$ |
| | | | 3 | 6 | $24, 25$ |
| | | | 5 | 12 | $26, 27$ |

In Table **??** "CS analysed" refers to the rightmost CS in the expression for the effect of the LIGR sequence, to which F is applied. In the column for the RCS's produced, the phrase "none useful" means that all the RCS's contain a c that would have to be passed to generate an LIGR (by Lemma **??**), therefore because this is impossible, there can be no new LIGR's derived by using any of these RCS's in the procedure for generating all the LIGR's. In the last column, the set of LIGR's produced is the same as the set of LIGR's produced from other CS's that follow the "from". These and the other results can be derived by using Lemmas **??**, **??**, and its corollary. In Table **??** the CS's to which F is applied head the two composite columns. The results for the LIGR's are given in the body of the table. The parameter $\mathtt{j}_1$ is as in Section **??**. The column headed # is the number of LIGR parts produced by the program "origins" [**?**] and the column headed LIGR's gives the identification numbers of the sets of LIGR's as defined in (**??**).

### 6.2.8 Sequences ending with LIGR $\geq 8$

LIGR 8 is $3\underline{\mathtt{T}} \leftarrow 3\underline{\mathtt{c}}\mathtt{T}$ and $3\alpha\underline{\mathtt{c}}\mathtt{T}\left\{\begin{matrix}\overset{\mathtt{b}}{\leftarrow} 2\underline{\mathtt{a}}\mathtt{cT}\\ \overset{\mathtt{c}}{\leftarrow} 3\underline{\mathtt{c}}\mathtt{cT}\end{matrix}\right.$ shortens to 4 and 8 so $\ldots 8 \overset{\mathtt{F}}{\rightarrow} \{4, 8\}$ but because the left end symbol of T is not specified, specialising it by including previous LIGR's could generate more results. $8 \cdot 8$ is $3\underline{\mathtt{T}} \leftarrow 3\underline{\mathtt{c}}\mathtt{T} \leftarrow 3\underline{\mathtt{c}}\mathtt{cT}$ and $3\alpha\underline{\mathtt{c}}\mathtt{cT}$ gives nothing new so $\ldots 8 \cdot 8 \overset{\mathtt{F}}{\rightarrow} \{4, 8\}$. Similarly it follows that $10 \cdot 8$ and $21 \cdot 8$ under F produce no new results, so $\ldots \{8, 10, 21\} \cdot 8 \overset{\mathtt{F}}{\rightarrow} \{4, 8\}$. In a similar manner also the following can be easily established $\ldots \{9\} \overset{\mathtt{F}}{\rightarrow} \{1, 5\}$ $\ldots \{10\} \overset{\mathtt{F}}{\rightarrow} \{4, 8\}$

$\ldots\{11\} \xrightarrow{F} \{3\}$
$\ldots\{12\} \xrightarrow{F} \{1,5\}$
$\ldots\{13\} \xrightarrow{F} \{3\}$
$\ldots\{14\} \xrightarrow{F} \{1,5\}$
In all these cases, F produces results that cannot, when specialised to the cases where T has particular forms, generate any new LIGR's from them. This results from the pointer not reaching next to the arbitrary string T at any point in these derivations. These are examples of the absence of RCS's in the result of F hence not giving any new LIGR's by specialising the original LIGR sequence by preceding it with another LIGR.

For sequences ending in $15-19$ the method given under sequences ending in 7 applies. The results for sequences ending $20, 21$ are just a single TM step in each case. For sequences ending in 22 and 23, the symbol a on the left makes these cases simple. For sequences ending in 25, after one reverse TM step Lemma **??** applies showing that no LIGR's can be generated regardless of any preceding LIGRs. For sequences ending in 26 and 27 the symbol a makes these analyses simple by Lemma **??** and likewise for sequences ending with $28-31$ the symbols c play the same role. All the results are in Table **??**.

### 6.2.9 Sequences ending with LIGR 24

LIGR 24 can be preceded by LIGR 7 so the set of all sequences ending in 7 can be each potentially followed by 24 (there could be cases where matching does not occur), so the same set of sequences ending in 7 in Table **??** are used initially. To these Lemma **??** is applied together with the use of the program "origins" [**?**]. These results are very similar to the set of all results for sequences ending in 7 with the result that usable RCS's occur except for the case $(7 \cdot 2)^3 \cdot 7 \cdot 24$ thus this is summed up in the shortest sequence of LIGR's that give no useable RCS's which is $(7 \cdot 2)^3 \cdot 7 \cdot 24$. The other cases should be regarded just as interim results and are not written in Table **??**. In all these cases the LIGR's produced are just $2, 22-25$. The next cases start with an LIGR not 2 or 7, and end with these alternating followed by 24 and have the same results. Next $18 \cdot 24$ gives no usable RCS's and again LIGR's $2, 22-25$. Next the grand search gives $24 \cdot 24$ followed by all sequences ending with 7 followed by $24 \cdot 24$. Again all these results are summarised by noting the shortest sequence not giving usable RCS's which is $(7 \cdot 2)^3 \cdot 7 \cdot 24 \cdot 24$ that generates only LIGR's $2, 22-25$ again. Next in the grand search is $18 \cdot 24 \cdot 24$ giving the same result again. Next is all the sequences ending with 7 followed by $24 \cdot 24 \cdot 24$. By now it looks as if an infinite sequence of similar results is occurring.

The reason for this is easy to find. Each time the LIGR 24 is added at the end of a sequence ending in 24, the number of c's in the string preceding the bdbd increases by 1. Replacing the d's by b to stop the computation

terminating, the initial steps in one branch of the backward search from $\mathtt{3cbbb\underline{b}}$ shows that

$$\mathtt{3T'cb^3\underline{b}}\alpha \leftarrow \begin{cases} \mathtt{1T'cbbbb\underline{b}} \\ \mathtt{2T'ccbab\underline{c}} \\ \mathtt{3T'ccbab\underline{d}} \\ \mathtt{3T'\underline{c}cdab}\alpha \\ \mathtt{2T'\underline{b}badb}\alpha \\ \mathtt{2T'ccbbb\underline{c}} \\ \mathtt{3T'ccbbb\underline{d}} \\ \mathtt{2T'\underline{b}bbcb}\alpha \\ \mathtt{2T'cbaab\underline{c}} \\ \mathtt{3T'cbaab\underline{d}} \\ \mathtt{2T'cbabb\underline{c}} \\ \mathtt{3T'cbabb\underline{d}} \end{cases} \tag{72}$$

giving LIGR's $2, 22-25$ and RCS's $\mathtt{3T'\underline{c}cdab}\alpha, \mathtt{2T'\underline{b}badb}\alpha, \mathtt{2T'\underline{b}bbcb}\alpha$. If the last symbol of $\mathtt{T'}$ is $\mathtt{c}$, continuing the backward search from these RCS's leads to one of the CS's $\mathtt{3c\underline{c}c}$ or $\mathtt{2c\underline{b}b}$. In each case the backward search has only one possible move given by $\mathtt{3c\underline{c}c} \leftarrow \mathtt{3\underline{c}cc}$ and $\mathtt{2c\underline{b}b} \leftarrow \mathtt{2\underline{b}bb}$ which can be iterated to give

$$\begin{aligned} \mathtt{3c^k\underline{c}c} &\leftarrow \mathtt{3\underline{c}c^{k+1}} \\ \mathtt{2c^k\underline{b}b} &\leftarrow \mathtt{2\underline{b}b^{k+1}} \end{aligned} \tag{73}$$

which applies if the last symbols of $\mathtt{T'}$ are $\mathtt{c}^k$. Therefore any RCS's resulting from applying $\mathtt{F}$ to any sequence ending with $\mathtt{24}$ are in 1 to 1 correspondence with any RCS's resulting from any sequence ending with $\mathtt{24}^{k+1}$ resulting from the insertion of either the string $\mathtt{c}^k$ or $\mathtt{b}^k$ according to (**??**) where $k$ is any positive integer. Such an RCS without any $\mathtt{c}$'s after the insertion (i.e. could generate new LIGR's) must correspond to a similar one before the insertion, i.e. RCS's for $k = \mathtt{1}$ which do not exist. Therefore there are no RCS's that could lead to new LIGR's for any $k > \mathtt{0}$.

Table 6: The RCS's and LIGR's resulting from $\mathtt{F}$ applied to sequences of LIGR's

| Possible sequences of LIGR's | It's effect | Usable RCS's i.e. those excluding those because of Lemmas **??** and **??** | LIGR's produced by $\mathtt{F}$ |
|---|---|---|---|
| $\mathtt{1}$ | $\mathtt{2\underline{T}} \overset{\mathtt{b}}{\leftarrow} \mathtt{1\underline{a}T}$ | $\mathtt{1\underline{c}aT}$ | $\mathtt{3}$ |
| $\mathtt{4 \cdot 1}$ | $\mathtt{3\underline{T}} \overset{\mathtt{b}}{\leftarrow} \mathtt{2\underline{a}T} \overset{\mathtt{b}}{\leftarrow} \mathtt{1\underline{a}aT}$ | $\mathtt{3}\alpha\mathtt{a\underline{d}T}$ | $\mathtt{3}$ |
| $\mathtt{8 \cdot 4 \cdot 1}$ | $\mathtt{3\underline{T}} \overset{\mathtt{c}}{\leftarrow} \mathtt{3\underline{c}T} \leftarrow \mathtt{1\underline{a}acT}$ | $\emptyset$ | $\mathtt{3}$ |
| $\mathtt{10 \cdot 4 \cdot 1}$ | $\mathtt{1\underline{c}aT} \leftarrow \mathtt{3\underline{c}cdT} \leftarrow \mathtt{1\underline{a}accdT}$ | $\emptyset$ | $\mathtt{3}$ |
| $\mathtt{21 \cdot 4 \cdot 1}$ | $\mathtt{1\underline{c}abcT} \leftarrow \mathtt{3\underline{c}ccdcT} \leftarrow \mathtt{1\underline{a}ac^3dcT}$ | $\emptyset$ | $\mathtt{3}$ |
| $\mathtt{5 \cdot 1}$ | $\mathtt{2\underline{T}} \leftarrow \mathtt{2\underline{b}T} \leftarrow \mathtt{1\underline{a}bT}$ | $\emptyset$ | $\mathtt{3}$ |
| $\mathtt{9 \cdot 1}$ | $\mathtt{1\underline{c}aT} \leftarrow \mathtt{2\underline{a}cdT} \leftarrow \mathtt{1\underline{a}acdT}$ | $\emptyset$ | $\mathtt{3}$ |
| $\mathtt{12 \cdot 1}$ | $\mathtt{1\underline{c}aaT} \leftarrow \mathtt{2\underline{b}badT} \leftarrow \mathtt{1\underline{a}bbadT}$ | $\emptyset$ | $\mathtt{3}$ |
| $\mathtt{14 \cdot 1}$ | $\mathtt{1\underline{c}cT} \leftarrow \mathtt{2\underline{b}bcT} \leftarrow \mathtt{1\underline{a}bbcT}$ | $\emptyset$ | $\mathtt{3}$ |

| | | | |
|---|---|---|---|
| $20 \cdot 1$ | $1\underline{c}abcT \leftarrow 2\underline{a}accdcT \leftarrow 1\underline{a}accdcT$ | $\emptyset$ | $3$ |
| $2$ | $3\underline{T} \overset{b}{\leftarrow} 1T\underline{b}$ | $\begin{cases} 3Tb\underline{d} \\ 2Tb\underline{c} \end{cases}$ | $\{6,7\}$ |
| $7a \cdot 2$ | $1\underline{T} \overset{a}{\leftarrow} 3T\underline{a} \leftarrow 1Ta\underline{b}$ | $\emptyset$ | $\{6,7\}$ |
| $7b \cdot 2$ | $1\underline{T} \overset{a}{\leftarrow} 3T\underline{b} \leftarrow 1Tb\underline{b}$ | $1T\underline{c}b\alpha$ | $\{6,7\}$ |
| $2 \cdot 7b \cdot 2$ | $3\underline{T} \overset{b}{\leftarrow} 1T\underline{b} \leftarrow 1Tbb\underline{b}$ | $\emptyset$ | $\{6,7\}$ |
| $15 \cdot 7 \cdot 2$ | $3Tb^5\underline{a} \leftarrow 1Tc\begin{Bmatrix} db \\ aa \end{Bmatrix}dbd\underline{b}$ $\leftarrow 1Tc\begin{Bmatrix} db \\ aa \end{Bmatrix}dbdbb\underline{b}$ | $\emptyset$ | $\{6,7\}$ |
| $16 \cdot 2$ | $3Tb^5\underline{a} \leftarrow 3Tc\begin{Bmatrix} db \\ aa \end{Bmatrix}dbdb\underline{b}$ $\leftarrow 1Tc\begin{Bmatrix} db \\ aa \end{Bmatrix}dbdbb\underline{b}$ | $\emptyset$ | $\{6,7\}$ |
| $18 \cdot 2$ | $3Tb^5\underline{a} \leftarrow 3Tcd\begin{Bmatrix} ba \\ cb \end{Bmatrix}db\underline{d} \leftarrow$ $1Tcd\begin{Bmatrix} ba \\ cb \end{Bmatrix}dbd\underline{b}$ | $\emptyset$ | $\{6,7\}$ |
| $\{22,24,26\} \cdot 2$ | $\leftarrow 1T^*dbd\underline{b}$ | $\emptyset$ | $\{6,7\}$ |
| $3$ | $1\underline{T} \overset{b}{\leftarrow} 1\underline{c}T$ | $1\underline{c}cT$ | $3$ |
| $1 \cdot 3$ | $2\underline{T} \leftarrow 1\underline{a}T \leftarrow 1\underline{c}aT$ | $3\alpha c\underline{d}T$ | $3$ |
| $4 \cdot 1 \cdot 3$ | $3\underline{T} \leftarrow 2\underline{a}T \leftarrow 1\underline{c}aaT$ | $3\alpha cb\underline{d}T$ | $\{3,9,10\}$ |
| $8 \cdot 4 \cdot 1 \cdot 3$ | $3\underline{T} \leftarrow 3\underline{c}T \leftarrow 1\underline{c}aacT$ | $\emptyset$ | $\{3,9,10,11,12\}$ |
| $10 \cdot 4 \cdot 1 \cdot 3$ | $1\underline{c}aT \leftarrow 3\underline{c}cdT$ $\leftarrow 1\underline{c}aaccdT$ | $\emptyset$ | $\{3,9,10,11,12\}$ |
| $21 \cdot 4 \cdot 1 \cdot 3$ | $1\underline{c}abcT \leftarrow 3\underline{c}ccdcT$ $\leftarrow 1\underline{c}aac^3dcT$ | $\emptyset$ | $\{3,9,10,11,12\}$ |
| $5 \cdot 1 \cdot 3$ | $2\underline{T} \leftarrow 2\underline{b}T \leftarrow 1\underline{c}abT$ | $1\alpha cd\underline{b}T$ | $\{3,9,10\}$ |
| $4 \cdot 5 \cdot 1 \cdot 3$ | $3\underline{T} \leftarrow 2\underline{a}T \leftarrow 1\underline{c}abaT$ | $3\alpha cdb\underline{d}T$ | $\{3,9,10\}$ |
| $8 \cdot 4 \cdot 5 \cdot 1 \cdot 3$ | $3\underline{T} \leftarrow 3\underline{c}T \leftarrow 1\underline{c}abacT$ | $\emptyset$ | $\{3,9,10\}$ |
| $10 \cdot 4 \cdot 5 \cdot 1 \cdot 3$ | $1\underline{c}aT \leftarrow 3\underline{c}cdT \leftarrow 1\underline{c}abaccdT$ | $\emptyset$ | $\{3,9,10\}$ |
| $21 \cdot 4 \cdot 5 \cdot 1 \cdot 3$ | $1\underline{c}abcT \leftarrow 3\underline{c}c^2dcT$ $\leftarrow 1\underline{c}abac^3dcT$ | $\emptyset$ | $\{3,9,10\}$ |
| $5 \cdot 5 \cdot 1 \cdot 3$ | $2\underline{T} \leftarrow 2\underline{b}T \leftarrow 1\underline{c}abbT$ | $\emptyset$ | $\{3,9,10\}$ |
| $9 \cdot 5 \cdot 1 \cdot 3$ | $1\underline{c}aT \leftarrow 2\underline{a}cdT \leftarrow 1\underline{c}abacdT$ | $\emptyset$ | $\{3,9,10\}$ |
| $12 \cdot 5 \cdot 1 \cdot 3$ | $1\underline{c}aaT \leftarrow 2\underline{b}badT \leftarrow 1\underline{c}ab^3adT$ | $\emptyset$ | $\{3,9,10\}$ |
| $14 \cdot 5 \cdot 1 \cdot 3$ | $1\underline{c}cT \leftarrow 2\underline{b}bcT \leftarrow 1\underline{c}ab^3cT$ | $\emptyset$ | $\{3,9,10\}$ |
| $20 \cdot 5 \cdot 1 \cdot 3$ | $1\underline{c}abcT \leftarrow 2\underline{a}accdcT$ $\leftarrow 1\underline{c}abaccdcT$ | $\emptyset$ | $\{3,9,10\}$ |
| $9 \cdot 1 \cdot 3$ | $1\underline{c}aT \leftarrow 2\underline{a}cdT \leftarrow 1\underline{c}aacdT$ | $\emptyset$ | $\{3,9,10,11,12\}$ |
| $12 \cdot 1 \cdot 3$ | $1\underline{c}aaT \leftarrow 2\underline{b}badT \leftarrow 1\underline{c}abbadT$ | $\emptyset$ | $\{3,9,10\}$ |
| $14 \cdot 1 \cdot 3$ | $1\underline{c}cT \leftarrow 2\underline{b}bcT \leftarrow 1\underline{c}abbcT$ | $\emptyset$ | $\{3,9,10\}$ |
| $20 \cdot 1 \cdot 3$ | $1\underline{c}abcT \leftarrow 2\underline{a}accdcT$ $\leftarrow 1\underline{c}aaccdcT$ | $\emptyset$ | $\{3,9,10,11,12\}$ |
| $3 \cdot 3$ | $1\underline{T} \overset{b}{\leftarrow} 1\underline{c}T \leftarrow 1\underline{c}cT$ | $2\alpha c\underline{c}T$ | $\{3\}$ |
| $1 \cdot 3 \cdot 3$ | $2\underline{T} \overset{b}{\leftarrow} 1\underline{a}T \leftarrow 1\underline{c}caT$ | $\emptyset$ | $\{3,13,14\}$ |
| $3 \cdot 3 \cdot 3$ | $1\underline{T} \overset{b}{\leftarrow} 1\underline{c}T \leftarrow 1\underline{c}ccT$ | $\emptyset$ | $\{3,13,14\}$ |
| $\{11,13,28,30\} \cdot 3 \cdot 3$ | $\leftarrow 1\underline{c}c\cdots$ | $\emptyset$ | $\{3\}$ |
| $11 \cdot 3$ | $1\underline{c}aaT \leftarrow 1\underline{a}baaT \leftarrow 1\underline{c}abaaT$ | $3\alpha cadb\underline{d}T$ | $\{3,9,10\}$ |
| $11 \cdot 3$ | $1\underline{c}aaT \leftarrow 1\underline{a}babT \leftarrow 1\underline{c}ababT$ | $1\alpha cdbd\underline{b}T$ | $\{3,9,10\}$ |
| $3 \cdot 11 \cdot 3$ | $1\underline{a}aT \leftarrow 1\underline{c}aaT \leftarrow 1\underline{c}abadT$ | Same results | $\{3,9,10\}$ |

| | | | |
|---|---|---|---|
| $1 \cdot 3 \cdot 11 \cdot 3$ | $2\underline{a}T \leftarrow 1\underline{a}aT \leftarrow 1\underline{c}abadT$ | Same results | $\{3, 9, 10\}$ |
| $4 \cdot 1 \cdot 3 \cdot 11 \cdot 3$ | $3\underline{T} \leftarrow 2\underline{a}T \leftarrow 1\underline{c}abadT$ | Same results | $\{3, 9, 10\}$ |
| $\{8, 10, 21\} \cdot 4 \cdot 1 \cdot 3 \cdot 11 \cdot 3$ | $\leftarrow 1\underline{c}abadcT$ | $\emptyset$ | $\{3, 9, 10\}$ |
| $5 \cdot 1 \cdot 3 \cdot 11 \cdot 3$ | No match | | |
| $9 \cdot 1 \cdot 3 \cdot 11 \cdot 3$ | $1\underline{c}aT \leftarrow 2\underline{a}cdT \leftarrow 1\underline{c}abaacdT$ | $\emptyset$ | $\{3, 9, 10\}$ |
| $9 \cdot 1 \cdot 3 \cdot 11 \cdot 3$ | $1\underline{c}aT \leftarrow 2\underline{a}cdT \leftarrow 1\underline{c}ababcdT$ | $\emptyset$ | $\{3, 9, 10, 28 - 31\}$ |
| $\{12, 14, 29, 31\} \cdot 1 \cdot 3 \cdot 11 \cdot 3$ | No match | | |
| $20 \cdot 1 \cdot 3 \cdot 11 \cdot 3$ | $1\underline{c}abcT \leftarrow 2\underline{a}ccdcT \leftarrow 1\underline{c}abadccdcT$ | $\emptyset$ | $\{3, 9, 10, 28 - 31\}$ |
| $3 \cdot 3 \cdot 11 \cdot 3$ | No match | | |
| $11 \cdot 3 \cdot 11 \cdot 3$ | No match | | |
| $13 \cdot 3 \cdot 11 \cdot 3$ | No match | | |
| $13 \cdot 3$ | $1\underline{c}cT \leftarrow 1\underline{a}bcT \leftarrow 1\underline{c}abcT$ | $2\alpha cdb\underline{c}T$ | $\{3, 9, 10\}$ |
| $3 \cdot 13 \cdot 3$ | $1\underline{c}T \leftarrow 1\underline{c}cT \leftarrow 1\underline{c}abcT$ | Same results | $\{3, 9, 10\}$ |
| $3 \cdot 3 \cdot 13 \cdot 3$ | $1\underline{T} \leftarrow 1\underline{c}T \leftarrow 1\underline{c}abcT$ | Same results | $\{3, 9, 10\}$ |
| $1 \cdot 3 \cdot 3 \cdot 13 \cdot 3$ | $2\underline{T} \leftarrow 1\underline{a}T \leftarrow 1\underline{c}abcaT$ | $\emptyset$ | $\{3, 9, 10, 20, 21\}$ |
| $3 \cdot 3 \cdot 3 \cdot 13 \cdot 3$ | $1\underline{T} \leftarrow 1\underline{c}T \leftarrow 1\underline{c}abccT$ | $\emptyset$ | $\{3, 9, 10, 20, 21\}$ |
| $11 \cdot 3 \cdot 3 \cdot 13 \cdot 3$ | $1\underline{c}aaT \leftarrow 1\underline{a}badT \leftarrow 1\underline{c}abcabadT$ | $\emptyset$ | $\{3, 9, 10, 20, 21\}$ |
| $13 \cdot 3 \cdot 3 \cdot 13 \cdot 3$ | $1\underline{c}cT \leftarrow 1\underline{a}bcT \leftarrow 1\underline{c}abcabcT$ | $\emptyset$ | $\{3, 9, 10, 20, 21\}$ |
| $\{11, 13\} \cdot 3 \cdot 13 \cdot 3$ | No match | | |
| $4$ | $3\underline{T} \leftarrow 2\underline{a}T$ | $\begin{cases} 1\underline{a}aT \\ 2\underline{b}aT \end{cases}$ | $\{1, 5\}$ |
| $8 \cdot 4$ | $3\underline{T} \leftarrow 3\underline{c}T \leftarrow 2\underline{a}cT$ | $\emptyset$ | $\{1, 5\}$ |
| $10 \cdot 4$ | $1\underline{c}aT \leftarrow 3\underline{c}cdT \leftarrow 2\underline{a}ccdT$ | $\emptyset$ | $\{1, 5\}$ |
| $21 \cdot 4$ | $1\underline{c}abcT \leftarrow 3\underline{c}ccdcT \leftarrow 2\underline{a}c^3dcT$ | $\emptyset$ | $\{1, 5\}$ |
| $5$ | $2\underline{T} \leftarrow 2\underline{b}T$ | $\begin{cases} 1\underline{a}bT \\ 2\underline{b}bT \end{cases}$ | $\{1, 5\}$ |
| $\{4, 5, 9, 12, 14, 20\} \cdot 5$ | $3\underline{T} \leftarrow 2\underline{a}T \leftarrow \begin{cases} 2\underline{b}a \ldots T \\ 2\underline{b}b \ldots T \end{cases}$ | $\emptyset$ | $\{1, 5\}$ |
| $6$ | $1\underline{T} \leftarrow 2T\underline{c}$ | $\emptyset$ | $\emptyset$ |
| $\{2, 15\} \cdot 6$ | $3\underline{T} \leftarrow 1T\underline{b} \leftarrow 2T \ldots b\underline{c}$ | $\emptyset$ | $\emptyset$ |
| $7$ | $1\underline{T} \leftarrow 3T\underline{d}$ | $1Td\underline{b}$ | $\{2\}$ |
| $2 \cdot 7$ | $3\underline{T} \leftarrow 1T\underline{b} \leftarrow 3Tb\underline{d}$ | $2T\underline{a}a\alpha$ | $\{2\}$ |
| $7 \cdot 2 \cdot 7$ | $1\underline{T} \leftarrow 3T\underline{d} \leftarrow 3Tdb\underline{d}$ | $1T\underline{a}ad\alpha$ | $\{2\}$ |
| $2 \cdot 7 \cdot 2 \cdot 7$ | $3\underline{T} \leftarrow 3Tbdb\underline{d}$ | $1T\underline{c}aad\alpha$ | $\{2, 22, 23\}$ |
| $7 \cdot 2 \cdot 7 \cdot 2 \cdot 7$ | $1\underline{T} \leftarrow 3Tdbdb\underline{d}$ | $1T\underline{a}badd\alpha$ | $\{2, 22 - 25\}$ |
| $2 \cdot 7 \cdot 2 \cdot 7 \cdot 2 \cdot 7$ | $3\underline{T} \leftarrow 3Tbdbdb\underline{d}$ | $1T\underline{c}abadd\alpha$ | $\{2, 22 - 25\}$ |
| $7 \cdot 2 \cdot 7 \cdot 2 \cdot 7 \cdot 2 \cdot 7$ | $1\underline{T} \leftarrow 3Tdbdbdb\underline{a}$ | $\emptyset$ | $\{2, 15 - 19\}$ |
| $7 \cdot 2 \cdot 7 \cdot 2 \cdot 7 \cdot 2 \cdot 7$ | $1\underline{T} \leftarrow 3Tdbdbdb\underline{b}$ | $\emptyset$ | $\{2, 22 - 27\}$ |
| $\{16, 18, 22, 24, 26\} \cdot (2 \cdot 7)^3$ | $\leftarrow 3T'(db)^3\underline{d}$ | $\emptyset$ | $\{2, 15 - 19, 22 - 27\}$ |
| $15 \cdot 7 \cdot 2 \cdot 7 \cdot 2 \cdot 7$ | $\leftarrow 3Tc\begin{Bmatrix} db \\ aa \end{Bmatrix}dbdbdbdb\underline{d}$ | $\emptyset$ | $\{2, 15 - 19, 22 - 27\}$ |
| $\{16, 18, 22, 24, 26\} \cdot 2 \cdot 7 \cdot 2 \cdot 7$ | $3Tb^5\underline{d} \leftarrow 3T'dbdbdb\underline{d}$ | $\emptyset$ | $\{2, 15 - 19, 22 - 27\}$ |
| $15 \cdot 7 \cdot 2 \cdot 7$ | $3Tb^5\underline{a} \leftarrow 3Tc\begin{Bmatrix} db \\ aa \end{Bmatrix}dbdbdb\underline{d}$ | $\emptyset$ | $\{2, 15 - 19, 22 - 27\}$ |
| $\{16, 22\} \cdot 2 \cdot 7$ | $\leftarrow 3T'adbdb\underline{a}$ | $\emptyset$ | $\{2\}$ |
| $\{16, 22\} \cdot 2 \cdot 7$ | $\leftarrow 3T'adbdb\underline{b}$ | $\emptyset$ | $\{2, 22 - 25\}$ |
| $18 \cdot 2 \cdot 7$<br>$24 \cdot 2 \cdot 7$<br>$26 \cdot 2 \cdot 7$ | $\left.\begin{array}{l} \leftarrow 3Tcd\begin{Bmatrix} ba \\ cb \end{Bmatrix}dbdb\underline{x} \\ \leftarrow 3Tcbdbdb\underline{x} \\ \leftarrow 3Tcadbdbdb\underline{x} \end{array}\right\}$ | $\emptyset$ | $\begin{cases} x = a & \{2, 15 - 19\} \\ x = b & \{2, 22 - 27\} \end{cases}$ |
| $15 \cdot 7$ | $3Tb^5\underline{a} \leftarrow 3Tc\begin{Bmatrix} db \\ aa \end{Bmatrix}dbdb\underline{d}$ | $\emptyset$ | $\{2, 15 - 19, 22 - 27\}$ |
| $8$ | $3\underline{T} \leftarrow 3\underline{c}T$ | $\begin{cases} 2\alpha\underline{a}cT \\ 3\alpha\underline{c}cT \end{cases}$ | $\{4, 8\}$ |
| $\{8, 10, 21\} \cdot 8$ | $\leftarrow 3\underline{c}cT'$ | $\emptyset$ | $\{4, 8\}$ |
| $9$ | $1\underline{c}aT \leftarrow 2\underline{a}cdT$ | $\emptyset$ | $\{1, 5\}$ |

| | | | |
|---|---|---|---|
| 10 | $1\underline{c}aT \leftarrow 3\underline{c}cdT$ | $\emptyset$ | $\{4,8\}$ |
| 11 | $1\underline{c}aaT \leftarrow 1\underline{a}badT$ | $\emptyset$ | $\{3\}$ |
| 12 | $1\underline{c}aaT \leftarrow 2\underline{b}badT$ | $\emptyset$ | $\{1,5\}$ |
| 13 | $1\underline{c}cT \leftarrow 1\underline{a}bcT$ | $\emptyset$ | $\{3\}$ |
| 14 | $1\underline{c}cT \leftarrow 2\underline{b}bcT$ | $\emptyset$ | $\{1,5\}$ |
| 15 | $3Tb^5\underline{a} \leftarrow 1Tc\begin{Bmatrix}db\\aa\end{Bmatrix}dbd\underline{b}$ | $\emptyset$ | $\{6,7\}$ |
| 16 | $3Tb^5\underline{a} \leftarrow 3Tca^3db\underline{d}$ | $\emptyset$ | $\{2,22,23\}$ |
| 17 | $3Tb^5\underline{a} \leftarrow 2Tca^3db\underline{c}$ | $\emptyset$ | $\emptyset$ |
| 18 | $3Tb^5\underline{a} \leftarrow 3Tcd\begin{Bmatrix}ba\\cb\end{Bmatrix}db\underline{d}$ | $\emptyset$ | $\{2,22-25\}$ |
| 19 | $3Tb^5\underline{a} \leftarrow 2Tcd\begin{Bmatrix}ba\\cb\end{Bmatrix}db\underline{c}$ | $\emptyset$ | $\emptyset$ |
| 20 | $1\underline{c}abcT \leftarrow 2\underline{a}ccdcT$ | $\emptyset$ | $\{1,5\}$ |
| 21 | $1\underline{c}abcT \leftarrow 3\underline{c}ccdcT$ | $\emptyset$ | $\{4,8\}$ |
| 22 | $3Tbb\underline{b} \leftarrow 3Tadb\underline{d}$ | $\emptyset$ | $\{2,22,23\}$ |
| 23 | $3Tbb\underline{b} \leftarrow 2Tadb\underline{c}$ | $\emptyset$ | $\emptyset$ |
| $(7\cdot 2)^3\cdot 7\cdot 24^k$<br>$\{16,18,22,24,26\}\cdot$<br>$(2\cdot 7)^3\cdot 24^k$<br>$15\cdot 7\cdot(2\cdot 7)^2\cdot 24^k$<br>$\{16,18,22,24,26\}\cdot$<br>$(2\cdot 7)^2\cdot 24^k$<br>$15\cdot 7\cdot 2\cdot 7\cdot 24^k$<br>$\{16,22\}\cdot$<br>$2\cdot 7\cdot 24^k$<br>$18\cdot 2\cdot 7\cdot 24^k$<br>$24\cdot 2\cdot 7\cdot 24^k$<br>$26\cdot 2\cdot 7\cdot 24^k$<br>$15\cdot 7\cdot 24^k$<br>$18\cdot 24^k$ | $\leftarrow 3Tdbdc^kbdb\underline{x}$<br>$\leftarrow 3Tdbdc^kbdb\underline{x}$<br>$\leftarrow 3Tcdbdbdbdc^kbdb\underline{x}$<br>$\leftarrow 3Tdbdc^kbdb\underline{x}$<br>$\leftarrow 3Tcdbdbdc^kbdb\underline{x}$<br>$\leftarrow 3Tadc^kbdb\underline{x}$<br>$\leftarrow 3Tcdcbdc^kbdb\underline{x}$<br>$\leftarrow 3Tcdcbdc^kbdb\underline{x}$<br>$\leftarrow 3Tcbdc^kbdb\underline{x}$<br>$\leftarrow 3Tcadbdc^kbdb\underline{x}$<br>$\leftarrow 3Tc\begin{Bmatrix}db\\aa\end{Bmatrix}dc^kbdb\underline{x}$ * | $\emptyset$ | $\begin{cases}x=a & \{2\}\\ x=b & \{2,22-25\}\end{cases}$ |
| 25 | $3Tb^3\underline{b} \leftarrow 2Tcbdb\underline{c}$ | $\emptyset$ | $\emptyset$ |
| 26 | $3Tb^5\underline{b} \leftarrow 3Tcadbdb\underline{d}$ | $\emptyset$ | $\{2,22-25\}$ |
| 27 | $3Tb^5\underline{b} \leftarrow 2Tcadbdb\underline{c}$ | $\emptyset$ | $\{2\}$ |
| 28 | $1\underline{a}bbccbT$ | $\emptyset$ | $\{3\}$ |
| 29 | $2\underline{b}bbccbT$ | $\emptyset$ | $\{1,5\}$ |
| 30 | $1\underline{a}bbccacT$ | $\emptyset$ | $\{3\}$ |
| 31 | $2\underline{b}bbccacT$ | $\emptyset$ | $\{1,5\}$ |

* See section **??**

********** checked to here ********** Section **??** has also had a lot of work done on it since the previous update.

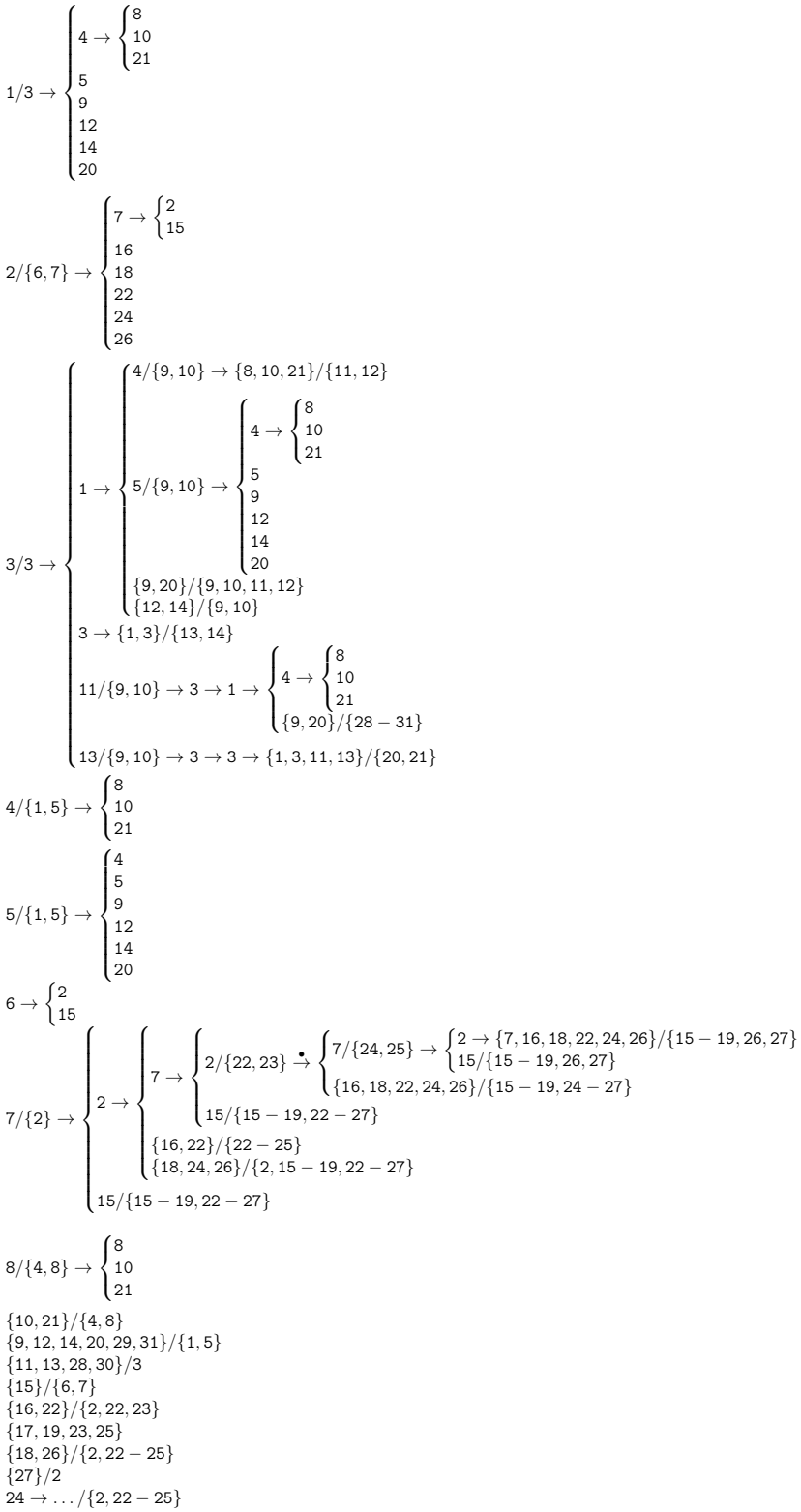


Figure 1: Map of the derivation of the LIGR's using the notation given in Section **??**. The bullet • means that the computation does not need to go beyond this point because the LIGR's generated are vacuous. See Section **??**.

## 6.3 Summary of the analysis for TM ??

In order to summarise further the results of the analysis of TM **??** in (**??**) just giving the LIGR's and showing the overall structure of the results, the following representation was developed that is given in Figure **??**. Next is a description of the notation used.

Each tree starts from one of the LIGR's $\mathtt{X_1}$ with $\mathtt{X_1}/\{\mathtt{Y_i}\} \to \mathtt{X_2}$, which means the LIGR $\mathtt{X_1}$ under $\mathtt{F}$ gives LIGR's $\{\mathtt{Y_i}\}$ and $\mathtt{X_2}$ is one of the LIGR's that can precede $\mathtt{X_1}$, and the sequence $\mathtt{S}$ of LIGR's starts with the single element $\mathtt{S} = \mathtt{X_1}$. The following appears repeatedly:
$\ldots \to \mathtt{X_1}/\{\mathtt{Y_i}\} \to \mathtt{X_2}/\{\mathtt{Y_{i+1}}\} \to \ldots$. This means $\mathtt{F}$ is applied to a sequence of LIGR's $\mathtt{S}$, that ends with $\mathtt{X_1}$, substituted into each other as above. This gives LIGR's $\Delta\mathtt{F} = \{\mathtt{Y_i}\}$ and in general some RCS's. There is in general a set of LIGR's $\mathtt{X_2}$ that can precede $\mathtt{X_1}$ and each such $\mathtt{X_2}$ defines a separate branch of the tree rooted by the original LIGR at the start. Branching points are indicated by braces. Applying $\mathtt{F}$ again starts recursively from each of the above RCS's with the substitution given by $\mathtt{X_2}$ i.e. $\mathtt{S}$ is replaced by $\mathtt{X_2} \cdot \mathtt{S}$ for each $\mathtt{X_2}$ that can precede $\mathtt{X_1}$. The result of this gives again the set of LIGR's $\Delta\mathtt{F_1}$ which is $\{\mathtt{Y_{i+1}}\}$ etc.. In general, there can be many members of $\mathtt{Y_i}$, $\mathtt{Y_{i+1}}$ etc. for each sequence $\mathtt{S}$. Also because $\mathtt{Y_i} = \emptyset$ is so frequent, $/\emptyset$ will be omitted if it occurs. A branch of the tree terminates when the set of RCS's above is $\emptyset$ or if there are no such LIGR's $\mathtt{X_2}$. Comparing **??** with (**??**) should clarify the notation.

Table of LIGR's is as follows obtained using the method of Section **??**. The column headed q is the length of the shortest IRR generated using it. If there isn't one the symbol $\infty$ is used making the LIGR vacuous.

| # | q | LIGR | # | q | LIGR |
|---|---|---|---|---|---|
| 1 | 2 | $2\underline{\mathtt{T}} \overset{b}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{T}$ | 17 | $\infty$ | $3\mathtt{Tb}^5\underline{\mathtt{a}} \overset{c}{\leftarrow} 2\mathtt{Tca}^3\mathtt{db}\underline{\mathtt{c}}$ |
| 2 | 2 | $3\underline{\mathtt{T}} \overset{b}{\leftarrow} 1\mathtt{T}\underline{\mathtt{b}}$ | 18 | $\infty$ | $3\mathtt{Tb}^5\underline{\mathtt{a}} \overset{a}{\leftarrow} 3\mathtt{Tcd}\left\{\begin{matrix}\mathtt{ba}\\ \mathtt{cb}\end{matrix}\right\}\mathtt{db}\underline{\mathtt{d}}$ |
| 3 | 2 | $1\underline{\mathtt{T}} \overset{b}{\leftarrow} 1\underline{\mathtt{c}}\mathtt{T}$ | 19 | $\infty$ | $3\mathtt{Tb}^5\underline{\mathtt{a}} \overset{c}{\leftarrow} 2\mathtt{Tcd}\left\{\begin{matrix}\mathtt{ba}\\ \mathtt{cb}\end{matrix}\right\}\mathtt{db}\underline{\mathtt{c}}$ |
| 4 | 2 | $3\underline{\mathtt{T}} \overset{b}{\leftarrow} 2\underline{\mathtt{a}}\mathtt{T}$ | 20 | 6 | $1\underline{\mathtt{c}}\mathtt{abcT} \overset{b}{\leftarrow} 2\underline{\mathtt{a}}\mathtt{ccdcT}$ |
| 5 | 2 | $2\underline{\mathtt{T}} \overset{c}{\leftarrow} 2\underline{\mathtt{b}}\mathtt{T}$ | 21 | 6 | $1\underline{\mathtt{c}}\mathtt{abcT} \overset{c}{\leftarrow} 3\underline{\mathtt{c}}\mathtt{ccdcT}$ |
| 6 | 2 | $1\underline{\mathtt{T}} \overset{c}{\leftarrow} 2\mathtt{T}\underline{\mathtt{c}}$ | 22 | 5 | $3\mathtt{Tbb}\underline{\mathtt{b}} \overset{a}{\leftarrow} 3\mathtt{Tadb}\underline{\mathtt{d}}$ |
| 7 | 2 | $1\underline{\mathtt{T}} \overset{a}{\leftarrow} 3\mathtt{T}\underline{\mathtt{d}}$ | 23 | 5 | $3\mathtt{Tbb}\underline{\mathtt{b}} \overset{c}{\leftarrow} 2\mathtt{Tadb}\underline{\mathtt{c}}$ |
| 8 | 2 | $3\underline{\mathtt{T}} \overset{c}{\leftarrow} 3\underline{\mathtt{c}}\mathtt{T}$ | 24 | $\infty$ | $3\mathtt{Tb}^3\underline{\mathtt{b}} \overset{a}{\leftarrow} 3\mathtt{Tcbdb}\underline{\mathtt{d}}$ |
| 9 | 4 | $1\underline{\mathtt{c}}\mathtt{aT} \overset{b}{\leftarrow} 2\underline{\mathtt{a}}\mathtt{cdT}$ | 25 | $\infty$ | $3\mathtt{Tb}^3\underline{\mathtt{b}} \overset{c}{\leftarrow} 2\mathtt{Tcbdb}\underline{\mathtt{c}}$ |
| 10 | 4 | $1\underline{\mathtt{c}}\mathtt{aT} \overset{c}{\leftarrow} 3\underline{\mathtt{c}}\mathtt{cdT}$ | 26 | $\infty$ | $3\mathtt{Tb}^5\underline{\mathtt{b}} \overset{a}{\leftarrow} 3\mathtt{Tcadbdb}\underline{\mathtt{d}}$ |
| 11 | 5 | $1\underline{\mathtt{c}}\mathtt{aaT} \overset{b}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{badT}$ | 27 | $\infty$ | $3\mathtt{Tb}^5\underline{\mathtt{b}} \overset{c}{\leftarrow} 2\mathtt{Tcadbdb}\underline{\mathtt{c}}$ |
| 12 | 5 | $1\underline{\mathtt{c}}\mathtt{aaT} \overset{c}{\leftarrow} 2\underline{\mathtt{b}}\mathtt{badT}$ | 28 | 7 | $1\underline{\mathtt{c}}\mathtt{ababT} \overset{b}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{bbccbT}$ |
| 13 | 4 | $1\underline{\mathtt{c}}\mathtt{cT} \overset{b}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{bcT}$ | 29 | 7 | $1\underline{\mathtt{c}}\mathtt{ababT} \overset{c}{\leftarrow} 2\underline{\mathtt{b}}\mathtt{bbccbT}$ |
| 14 | 4 | $1\underline{\mathtt{c}}\mathtt{cT} \overset{c}{\leftarrow} 2\underline{\mathtt{b}}\mathtt{bcT}$ | 30 | 9 | $1\underline{\mathtt{c}}\mathtt{ababcT} \overset{b}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{bbccacT}$ |
| 15 | $\infty$ | $3\mathtt{Tb}^5\underline{\mathtt{a}} \overset{b}{\leftarrow} 1\mathtt{Tc}\left\{\begin{matrix}\mathtt{db}\\ \mathtt{aa}\end{matrix}\right\}\mathtt{dbd}\underline{\mathtt{b}}$ | 31 | 9 | $1\underline{\mathtt{c}}\mathtt{ababcT} \overset{c}{\leftarrow} 2\underline{\mathtt{b}}\mathtt{bbccacT}$ |
| 16 | $\infty$ | $3\mathtt{Tb}^5\underline{\mathtt{a}} \overset{a}{\leftarrow} 3\mathtt{Tca}^3\mathtt{db}\underline{\mathtt{d}}$ | | | |

(74)

Because new LIGR's have been found, the "can be preceded by" relation needs to be updated as follows and relates to the numbering of LIGR's in (**??**).

| | |
|---|---|
| 1, 5 | 4, 5, 9, 12, 14, 20, 29, 31 |
| 2 | 7, 16, 18, 22, 24, 26 |
| 3 | 1, 3, 11, 13, 28, 30 |
| 4, 8 | 8, 10, 21 |
| 6, 7 | 2, 15 |
| $9-14, 20, 21$ | 3 |
| 15, 16, 17, 18, 19 | 7 |
| 20, 21 | 3 |
| 22, 23 | 7, 16, 22, 24, 26 |
| 24, 25 | 7, 18, 24, 26 |
| 26, 27 | 7 |
| 28, 29, 30, 31 | 3 |

(75)

Note that there may be other symbols in either of the strings $\mathtt{T}$ that prevent a match of the sequences.

The results summarised in Table **??** will show that (when complete and correct) the set of LIGR's $\mathtt{1}-\mathtt{31}$ is closed under $\mathtt{F}$ and therefore by Theorem **??** these are sufficient to derive all the IRR's from the IRR(2).

# 7 Return to the IGR's and their significance

The IGR's in Table **??** can be expressed in a more organised way in the Table **??**. Importantly the LIGR's involved on the left hand side and obtained from the computer calculation, agree with the very long calculation in the preceding section summarised in Table **??**.

The following is the table of the distinct RIGR's in Table **??** with the $\alpha$ values put back in and the redundant symbols $\mathtt{T_2}$ representing arbitrary strings removed.

Table 7: The set of RIGR's

| | | | | |
|---|---|---|---|---|
| $\mathtt{1\underline{a} \to 2b\_}$ | $\mathtt{1\underline{b} \to 3\_b}$ | $\mathtt{1\underline{c} \to 1b\_}$ | $\mathtt{1\underline{c}a \to 2bb\_}$ | $\mathtt{1\underline{c}abab \to 3b^5\_}$ |
| $\mathtt{1\underline{c}ababa \to 3\_bababa}$ | $\mathtt{1\underline{c}ababc \to 3bbbbbc\_}$ | $\mathtt{1\underline{c}abc \to 1bbbb\_}$ | $\mathtt{1\underline{c}c \to 1bb\_}$ | $\mathtt{2\underline{a} \to 3b\_}$ |
| $\mathtt{2\underline{b} \to 2c\_}$ | $\mathtt{2b\underline{c} \to 3\_bc}$ | $\mathtt{2bb\underline{c} \to 1\_abc}$ | $\mathtt{2c\underline{c} \to 1bb\_}$ | $\mathtt{3b^5\underline{a} \to 3\_bababa}$ |
| $\mathtt{3cb\underline{a} \to 3bbb\_}$ | $\mathtt{3\underline{b} \to 1\_a}$ | $\mathtt{3bb\underline{b} \to 1\_aba}$ | $\mathtt{3bbb\underline{b} \to 3\_baba}$ | $\mathtt{3bbbbb\underline{b} \to 3\_bababa}$ |
| $\mathtt{3cb\underline{b} \to 3bbb\_}$ | $\mathtt{3c\underline{b} \to 2bb\_}$ | $\mathtt{3\underline{c} \to 3c\_}$ | $\mathtt{3\underline{c}ba \to 3bbb\_}$ | $\mathtt{3\underline{c}bab \to 3\_baba}$ |

Table 8: The table of IGR's (Table **??**) re-expressed in terms of LIGR's and RIGR's

| $\alpha$ | LIGR | RIGR |
|---|---|---|
| b | $1\underline{T_1} \leftarrow 1\underline{c}T_1$<br>$2\underline{T_1} \leftarrow 1\underline{a}T_1$<br>$3\underline{T_1} \leftarrow 2\underline{a}T_1$<br>$1\underline{c}aT_1 \leftarrow 2\underline{a}cdT_1$<br>$1\underline{c}cT_1 \leftarrow 1\underline{a}bcT_1$<br>$1\underline{c}aaT_1 \leftarrow 1\underline{a}badT_1$<br>$1\underline{c}abcT_1 \leftarrow 2\underline{a}ccdcT_1$<br>$1\underline{c}ababT_1 \leftarrow 1\underline{a}bbccbT_1$<br>$1\underline{c}ababcT_1 \leftarrow 1\underline{a}bbccacT_1$ | $1\_T_2 \rightarrow 3\_bT_2$<br>$3\_T_2 \rightarrow 1\_aT_2$ |
| c | $3\underline{T_1} \leftarrow 3\underline{c}T_1$ | $1\_aT_2 \rightarrow 2bb\underline{T_2}$<br>$3\_babT_2 \rightarrow 3\_baba\underline{T_2}$ |
| c | $2\underline{T_1} \leftarrow 2\underline{b}T_1$ | $1\_cT_2 \rightarrow 1bb\underline{T_2}$<br>$3\_baT_2 \rightarrow 3bbb\underline{T_2}$<br>$3\_babT_2 \rightarrow 3\_baba\underline{T_2}$<br>$1\_ababaT_2 \rightarrow 3\_bababa\underline{T_2}$ |
| a | $1\underline{T_1} \leftarrow 3T_1\underline{d}$ | $1T_{2\_} \rightarrow 2T_2b\_$<br>$2T_{2\_} \rightarrow 3T_2b\_$<br>$3T_2b^5\_ \rightarrow 3\underline{T_2}bababa$ |
| c | $1\underline{T_1} \leftarrow 2T_1\underline{c}$ | $1T_{2\_} \rightarrow 1T_2b\_$<br>$3T_{2\_} \rightarrow 3T_2c\_$<br>$2T_2b\_ \rightarrow 3\underline{T_2}bc$<br>$2T_2c\_ \rightarrow 1\underline{T_2}bb\_$<br>$2T_2bb\_ \rightarrow 1\underline{T_2}abc$ |
| b | $3\underline{T_1} \leftarrow 1T_1\underline{b}$ | $2T_{2\_} \rightarrow 2T_2c\_$<br>$3T_2c\_ \rightarrow 2T_2bb\_$<br>$3T_2bb\_ \rightarrow 1\underline{T_2}aba$<br>$3T_2cb\_ \rightarrow 3\underline{T_2}bbb\_$<br>$3T_2bbb\_ \rightarrow 3\underline{T_2}baba$<br>$3T_2bbbbb\_ \rightarrow 3\underline{T_2}bababa$ |
| c | $1\underline{c}cT_1 \leftarrow 2\underline{b}bcT_1$ | $3\_babT_2 \rightarrow 3\_baba\underline{T_2}$<br>$1\_ababT_2 \rightarrow 3b^5\underline{T_2}$<br>$1\_ababaT_2 \rightarrow 3\_bababa\underline{T_2}$<br>$1\_ababcT_2 \rightarrow 3bbbbbc\underline{T_2}$ |
| c | $1\underline{c}aT_1 \leftarrow 3\underline{c}cdT_1$ | $1\_abcT_2 \rightarrow 1bbbb\underline{T_2}$<br>$3\_babT_2 \rightarrow 3\_baba\underline{T_2}$<br>$1\_ababaT_2 \rightarrow 3\_bababa\underline{T_2}$ |
| c | $1\underline{c}aaT_1 \leftarrow 2\underline{b}badT_1$<br>$1\underline{c}abcT_1 \leftarrow 3\underline{c}ccdcT_1$<br>$1\underline{c}ababT_1 \leftarrow 2\underline{b}bbccbT_1$<br>$1\underline{c}ababcT_1 \leftarrow 2\underline{b}bbccacT_1$ | $3\_babT_2 \rightarrow 3\_baba\underline{T_2}$<br>$1\_ababaT_2 \rightarrow 3\_bababa\underline{T_2}$ |
| c | $3T_1bb\underline{b} \leftarrow 2T_1adb\underline{c}$ | $3T_{2\_} \rightarrow 3T_2c\_$ |
| a | $3T_1bb\underline{b} \leftarrow 3T_1adb\underline{d}$ | $3T_2cb\_ \rightarrow 3T_2bbb\_$<br>$3T_2b^5\_ \rightarrow 3\underline{T_2}bababa$ |

Notice that the complete Table **??** consisting of 11 blocks has the property

that within each block every LIGR combines with every RIGR to form an IGR. Here the LIGR is $\texttt{CS1} \leftarrow \texttt{CS3}$ and the RIGR is $\texttt{CS2} \rightarrow \texttt{CS4}$ i.e. the IGR is $\texttt{CS1} \rightarrow\!\!\rightarrow \texttt{CS2} \Rightarrow \texttt{CS3} \rightarrow\!\!\rightarrow \texttt{CS4}$ provided $\texttt{CS1} \rightarrow\!\!\rightarrow \texttt{CS2}$ has the type LRL or RLR. The last condition is automatically satisfied because within each block in Table **??** CS1 and CS2 both have the pointer on the same side (and strings $\texttt{T}_1, \texttt{T}_2$).

The result with IGR's $\texttt{4}, \texttt{5}, \texttt{14}, \texttt{15}$ omitted (corresponding to 8.1 and 9.1 in block 1, and 3.2 and 4.2 in block 9 of Table **??**) occurred when the program was run with $n = 9$. In this case the incomplete Table **??** does not have the above property.

Can the IGR's be obtained directly (based on applying the process called F directly starting from the single TM steps) or based on the LIGR's without first obtaining the IRR's to a length unknown beforehand as was done in the computer program [**?**]? Note that this computer algorithm is very inefficient.

It follows from the fact that each IGR has two independent parts (its LIGR and its RIGR) once the value of $\alpha$ is fixed that if w and x are LIGR's, and y and z are RIGRs, all with the same value of $\alpha$ that

$$(\texttt{w}, \texttt{y}), (\texttt{x}, \texttt{y}), (\texttt{w}, \texttt{z}) \in \texttt{IGR} \Rightarrow (\texttt{x}, \texttt{z}) \in \texttt{IGR} \tag{76}$$

again with the same value of $\alpha$ and where the ordered pairs of an LIGR and an RIGR are combined to give IGR's as above i.e.

**Theorem 7.1.** *given that* $(\texttt{w}, \texttt{y}) \in \texttt{IGR}$*, any LIGR* x *such that* $(\texttt{x}, \texttt{y}) \in \texttt{IGR}$ *can be paired with any RIGR* z *such that* $(\texttt{w}, \texttt{z}) \in \texttt{IGR}$ *to give another result* $(\texttt{x}, \texttt{z}) \in \texttt{IGR}$*, again with the same value of* $\alpha$*, thus the structure of Table* **??** *is general.*

## 7.1 Attempts to find all the IGR's using a method similar to the way the LIGR's were found.

In this method all possible sequences of IGR's currently known are considered, starting with all the IGR's needed to obtain the set IRR(2) which are labelled $\texttt{1} - \texttt{8}$ in (**??**). They are combined with $\cdot$ defined above, and F is applied to the resulting IRR pattern on its RHS to generate a new IGR (after removing redundant symbols). At the same time the table of the relation "can possibly be preceded by" (**??**) for IGR's is kept up-to-date to facilitate this. The summary of the results is given in Figure **??** where the IGR's $\texttt{9} - \texttt{22}$ are derived by this method but the calculation came to an end when it was clearly incomplete with only 25 IGR's out of the 55 obtained in Table **??** with not all the LIGR's in (**??**) being involved. This is related to the fact that there were no IGR's that could precede IGR's $\texttt{1}, \texttt{4}, \texttt{5}$ in (**??**).

$$
\begin{array}{ll}
1 & \text{context } \{(\mathtt{a},\mathtt{a}),(\mathtt{b},\mathtt{a})\}3\underline{\mathtt{T}_1} \to\!\to 1\_\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 2\underline{\mathtt{a}}\mathtt{T}_1 \to\!\to 3\_\mathtt{b}\mathtt{T}_2 \\
2 & \text{context } \{(\mathtt{c},\mathtt{c})\}2\underline{\mathtt{T}_1} \to\!\to 1\_\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 1\underline{\mathtt{a}}\mathtt{T}_1 \to\!\to 3\_\mathtt{b}\mathtt{T}_2 \\
3 & \text{context } \{(\mathtt{b},\mathtt{b})\}1\underline{\mathtt{T}_1} \to\!\to 3\_\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 1\underline{\mathtt{c}}\mathtt{T}_1 \to\!\to 1\_\mathtt{a}\mathtt{T}_2 \\
4,5 & \text{context } \{(\mathtt{c},\mathtt{b})\}1\underline{\mathtt{T}_1} \to\!\to 1\mathtt{T}_2\_ \begin{cases} \overset{\mathtt{a}}{\Rightarrow} 3\mathtt{T}_1\underline{\mathtt{d}} \to\!\to 2\mathtt{T}_2\mathtt{b}\_ \\ \overset{\mathtt{c}}{\Rightarrow} 2\mathtt{T}_1\underline{\mathtt{c}} \to\!\to 1\mathtt{T}_2\mathtt{b}\_ \end{cases} \\
6 & \text{context } \{(\mathtt{a},\mathtt{b})\}1\underline{\mathtt{T}_1} \to\!\to 2\mathtt{T}_2\_ \overset{\mathtt{a}}{\Rightarrow} 3\mathtt{T}_1\underline{\mathtt{d}} \to\!\to 3\mathtt{T}_2\mathtt{b}\_ \\
7 & \text{context } \{(\mathtt{c},)\}3\underline{\mathtt{T}_1} \to\!\to 3\mathtt{T}_2\mathtt{c}\_ \overset{\mathtt{b}}{\Rightarrow} 1\mathtt{T}_1\underline{\mathtt{b}} \to\!\to 2\mathtt{T}_2\mathtt{b}\mathtt{b}\_ \\
8 & \text{context } \{(\mathtt{a},),(\mathtt{b},)\}3\underline{\mathtt{T}_1} \to\!\to 1\_\mathtt{a}\mathtt{T}_2 \overset{\mathtt{c}}{\Rightarrow} 3\underline{\mathtt{c}}\mathtt{T}_1 \to\!\to 2\mathtt{b}\mathtt{b}\underline{\mathtt{T}_2} \\
\mathtt{a} & \text{context } \{(\mathtt{c},)\}2\underline{\mathtt{T}_1} \to\!\to 1\_\mathtt{c}\mathtt{T}_2 \overset{\mathtt{c}}{\Rightarrow} 2\underline{\mathtt{b}}\mathtt{T}_1 \to\!\to 1\mathtt{b}\mathtt{b}\underline{\mathtt{T}_2} \\
\mathtt{b} & \text{context } \{(\mathtt{a},)\}1\underline{\mathtt{T}_1} \to\!\to 2\mathtt{T}_2\mathtt{b}\_ \overset{\mathtt{c}}{\Rightarrow} 2\mathtt{T}_1\underline{\mathtt{c}} \to\!\to 3\underline{\mathtt{T}_2}\mathtt{b}\mathtt{c} \\
\mathtt{b}' & 1\underline{\mathtt{T}_1} \to\!\to 2\mathtt{T}_2\mathtt{b}\mathtt{b}\_ \overset{\mathtt{c}}{\Rightarrow} 2\mathtt{T}_1\underline{\mathtt{c}} \to\!\to 1\underline{\mathtt{T}_2}\mathtt{a}\mathtt{b}\mathtt{c} \\
9 & 2\underline{\mathtt{T}_1} \to\!\to 3\_\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 1\underline{\mathtt{a}}\mathtt{T}_1 \to\!\to 1\_\mathtt{a}\mathtt{T}_2 \\
10 & 2\underline{\mathtt{T}_1} \to\!\to 3\_\mathtt{b}\mathtt{T}_2 \overset{\mathtt{c}}{\Rightarrow} 2\underline{\mathtt{b}}\mathtt{T}_1 \to\!\to 2\mathtt{b}\mathtt{b}\underline{\mathtt{T}_2} \\
10' & 2\underline{\mathtt{T}_1} \to\!\to 3\_\mathtt{b}\mathtt{a}\mathtt{T}_2 \overset{\mathtt{c}}{\Rightarrow} 2\underline{\mathtt{b}}\mathtt{T}_1 \to\!\to 3\mathtt{b}\mathtt{b}\mathtt{b}\underline{\mathtt{T}_2} \\
11 & 1\underline{\mathtt{T}_1} \to\!\to 1\_\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 1\underline{\mathtt{c}}\mathtt{T}_1 \to\!\to 3\_\mathtt{b}\mathtt{T}_2 \\
12 & 3\underline{\mathtt{T}_1} \to\!\to 2\mathtt{T}_2\_ \overset{\mathtt{b}}{\Rightarrow} 1\mathtt{T}_1\underline{\mathtt{b}} \to\!\to 2\mathtt{T}_2\mathtt{c}\_ \\
13 & 3\underline{\mathtt{T}_1} \to\!\to 3\mathtt{T}_2\mathtt{b}\_ \overset{\mathtt{b}}{\Rightarrow} 1\mathtt{T}_1\underline{\mathtt{b}} \to\!\to 3\underline{\mathtt{T}_2}\mathtt{b}\mathtt{a} \\
13' & 3\underline{\mathtt{T}_1} \to\!\to 3\mathtt{T}_2\mathtt{b}\mathtt{b}\_ \overset{\mathtt{b}}{\Rightarrow} 1\mathtt{T}_1\underline{\mathtt{b}} \to\!\to 1\underline{\mathtt{T}_2}\mathtt{a}\mathtt{b}\mathtt{a} \\
14 & 3\underline{\mathtt{T}_1} \to\!\to 3\mathtt{T}_2\mathtt{b}^3\_ \overset{\mathtt{b}}{\Rightarrow} 1\mathtt{T}_1\underline{\mathtt{b}} \to\!\to 3\underline{\mathtt{T}_2}\mathtt{b}\mathtt{a}\mathtt{b}\mathtt{a} \\
15 & 1\underline{\mathtt{c}}\mathtt{a}\mathtt{T}_1 \to\!\to 3\_\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 2\underline{\mathtt{a}}\mathtt{c}\mathtt{d}\mathtt{T}_1 \to\!\to 1\_\mathtt{a}\mathtt{T}_2 \\
16 & 1\underline{\mathtt{T}_1} \to\!\to 2\mathtt{T}_2\mathtt{c}\_ \overset{\mathtt{c}}{\Rightarrow} 2\mathtt{T}_1\underline{\mathtt{c}} \to\!\to 1\mathtt{T}_2\mathtt{b}\mathtt{b}\_ \\
17 & 2\underline{\mathtt{T}_1} \to\!\to 1\_\mathtt{a}\mathtt{T}_2 \overset{\mathtt{c}}{\Rightarrow} 2\underline{\mathtt{b}}\mathtt{T}_1 \to\!\to 2\mathtt{b}\mathtt{b}\underline{\mathtt{T}_2} \\
18 & 3\underline{\mathtt{T}_1} \to\!\to 3\mathtt{T}_2\mathtt{c}\mathtt{b}\_ \overset{\mathtt{b}}{\Rightarrow} 1\mathtt{T}_1\underline{\mathtt{b}} \to\!\to 3\mathtt{T}_2\mathtt{b}\mathtt{b}\mathtt{b}\_ \\
19 & 1\underline{\mathtt{T}_1} \to\!\to 3\mathtt{T}_2\mathtt{b}^3\_ \overset{\mathtt{a}}{\Rightarrow} 3\mathtt{T}_1\underline{\mathtt{d}} \to\!\to 3\underline{\mathtt{T}_2}\mathtt{b}\mathtt{a}\mathtt{b}\mathtt{a} \\
20 & 1\underline{\mathtt{T}_1} \to\!\to 3\mathtt{T}_2\_ \overset{\mathtt{c}}{\Rightarrow} 2\mathtt{T}_1\underline{\mathtt{c}} \to\!\to 3\mathtt{T}_2\mathtt{c}\_ \\
21 & 1\underline{\mathtt{c}}\mathtt{a}\mathtt{a}\mathtt{T}_1 \to\!\to 1\_\mathtt{T}_2 \overset{\mathtt{b}}{\Rightarrow} 1\underline{\mathtt{a}}\mathtt{b}\mathtt{a}\mathtt{d}\mathtt{T}_1 \to\!\to 3\_\mathtt{b}\mathtt{T}_2 \\
22 & 1\underline{\mathtt{c}}\mathtt{a}\mathtt{T}_1 \to\!\to 1\_\mathtt{a}\mathtt{b}\mathtt{a}\mathtt{T}_2 \overset{\mathtt{c}}{\Rightarrow} 3\underline{\mathtt{c}}\mathtt{c}\mathtt{d}\mathtt{T}_1 \to\!\to 3\mathtt{b}\mathtt{b}\mathtt{c}\mathtt{b}\underline{\mathtt{T}_2}
\end{array}
\tag{77}
$$

Later it became clear that the contexts for the IGR's generating the IRR(2) are needed as well and the results of this second attempt are also shown here. This resulted in IGR's a, b and b′, 10′ and 13′ being generated as well as IGR's $9-22$ with the exception of IGR's 10 and 13 that only belonged to the first

$1/\{9, 10'\}$.
$2/3 \to 15$.
$3/11$.
$4/12 \to$
$5 \to$
$6/13'$.
$7/\{6, \mathtt{b}'\}$.
8.
a.
b.
$9/11 \to 1 \to$
10.
$11/3 \to 9 \to 1/15 \to$
$12/\{6, 16\} \to 4 \to$
$13'$.
14.
$15/\{2, 17\}$.
$16 \to 12 \to 4$
17.
$18/\{19, 20\} \to 6 \to 12 \to 4$. $* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *$
19.
20.
$21/3$.
22.

Figure 2: Map of the derivation of the IGR's using notation similar to that of Figure **??**. A full stop means that no RCS's are present. Thus no other preceding IGR's can give any more results.

attempt. The relation "can possibly be preceded by" is as follows

| | |
|---|---|
| 2 | 15 |
| 3 | 2, 11, 18, 21 |
| 6 | 7, 12 |
| 9 | 1, 20 |
| $10'$ | 1 |
| 11 | 3, 9 |
| 12 | 4 |
| $13'$ | 6 |
| 14 | 6 |
| 15 | 3, 11 |
| 16 | 12 |
| 17 | 15 |
| 18 | 6 |
| 19 | 18 |
| 20 | 18 |
| 21 | 3 |
| 22 | 3 |

(78)

## 7.2 More on the relationships between IRR's, IGR's and LIGR's

In this section, methods are illustrated for (1) finding IGR's having a given LIGR, (2) checking that LIGR's are not vacuous, which can lead to an alternative descriptions of the behaviour of a TM. The discrepancy between the LIGR's associated with **??** and **??** for TM (**??**) is resolved.

Consider the derivation of IGR 7 of Table **??** i.e.

$$1\underline{c}cT_1 \rightarrow\rightarrow 1\_T_2 \overset{b}{\Rightarrow} 1\underline{a}bcT_1 \rightarrow\rightarrow 3\_bT_2 \tag{79}$$

This IGR first appears in the table of IRR's with the context (cb, abab) giving the full form $1\underline{c}ccbT_1 \rightarrow\rightarrow 1\_ababT_2 \overset{b}{\Rightarrow} 1\underline{a}bccbT_1 \rightarrow\rightarrow 3\_bababT_2$, which can be derived by successively applying F giving the following sequence of IRR patterns which can be traced back using $F^{-1}$ described at the end of Section **??**:

$$\begin{array}{l} 1\underline{b}T_1 \rightarrow\rightarrow 3\_bT_2 \\ 1\underline{c}bT_1 \rightarrow\rightarrow 1\_abT_2 \\ 1\underline{c}cbT_1 \rightarrow\rightarrow 3\_babT_2 \\ 1\underline{c}ccbT_1 \rightarrow\rightarrow 1\_ababT_2 \\ 1\underline{a}bccbT_1 \rightarrow\rightarrow 3\_bababT_2 \end{array} \tag{80}$$

The non-redundant forms of the IGR's for these steps are respectively $1\underline{T_1} \rightarrow\rightarrow 3\_T_2 \overset{b}{\Rightarrow} 1\underline{c}T \rightarrow\rightarrow 1\_aT_2$ with context (b, b) i.e. IGR 3, $1\underline{T_1} \rightarrow\rightarrow 1\_T_2 \overset{b}{\Rightarrow}$

$\mathtt{1\underline{c}T} \rightarrow\!\!\!\rightarrow \mathtt{3\_bT_2}$ with context $(\mathtt{cb}, \mathtt{ab})$ and IGR 3 again with context $(\mathtt{ccb}, \mathtt{bab})$, and (**??**) with context $(\mathtt{cb}, \mathtt{abab})$. Note that (**??**) cannot be derived from these reduced forms unless these contexts are known beforehand. That is, (**??**) can only be found once the IRR pattern $\mathtt{1\underline{c}ccbT_1} \rightarrow\!\!\!\rightarrow \mathtt{1\_ababT_2}$ has been found in the IRR's.

Consider another example. When an origin that starts with $\mathtt{1\underline{c}c}$ has been found in an IRR pattern such as $\mathtt{1\underline{c}cbT_1} \rightarrow\!\!\!\rightarrow \mathtt{3\_babT_2}$. Then F can be applied to this giving $\left.\begin{matrix}\mathtt{1\underline{c}ccb}\\ \mathtt{1\underline{a}bcb}\end{matrix}\right\}\mathtt{T_1} \rightarrow\!\!\!\rightarrow \mathtt{1\underline{a}babT_2}$ and $\mathtt{2\underline{b}bcbT_1} \rightarrow\!\!\!\rightarrow \mathtt{3\_babaT_2}$ with $\alpha =$ b, c respectively. The results are reduced to minimal form by removing symbols that are not involved in the computation, then the original string might remain intact. In this case it gives the IGR's

$$\begin{array}{l}\mathtt{1\underline{T_1}} \rightarrow\!\!\!\rightarrow \mathtt{3\_T_2} \overset{\mathtt{b}}{\Rightarrow} \mathtt{1\underline{c}T_1} \rightarrow\!\!\!\rightarrow \mathtt{1\_aT_2}\\ \mathtt{1\underline{c}cT_1} \rightarrow\!\!\!\rightarrow \mathtt{3\_T_2} \overset{\mathtt{b}}{\Rightarrow} \mathtt{1\underline{a}bcT_1} \rightarrow\!\!\!\rightarrow \mathtt{1\_aT_2}\\ \mathtt{1\underline{c}cT_1} \rightarrow\!\!\!\rightarrow \mathtt{3\_babT_2} \overset{\mathtt{c}}{\Rightarrow} \mathtt{2\underline{b}bcT_1} \rightarrow\!\!\!\rightarrow \mathtt{3\_babaT_2}\end{array} . \tag{81}$$

two of which contain the $\mathtt{1\underline{c}c}$ on the left. Thus this method can generate IGR's that have a given LIGR by taking its LHS at the start.

A very similar approach can be used to check that all the LIGR's in (**??**) have matching IGR's from which it can easily be checked whether or not they are actually used in IRR's, because some of them were not verified in the computer output in Table **??**. These were the LIGR's (after knocking off the arbitrary string T) with LHS's

$$\mathtt{3b^5\underline{a}}, \mathtt{3b^5\underline{b}}, \mathtt{3b^3\underline{b}}. \tag{82}$$

To do this two definitions will simplify the presentation.

**Definition 7.2.** *A logically valid IGR is non-vacuous if and only if there is a combination of the symbol strings* $\mathtt{T_1}$ *and* $\mathtt{T_2}$ *that makes it match an IRR of extendable type of the TM in its LHS and an IRR of the TM on its RHS.*

If this is not true it means that the IGR is in practical terms useless although it is still a valid logical statement. Examples will be found below.

**Definition 7.3.** *An LIGR is non-vacuous if and only if there is an IGR that contains it (in the sense of Section* **??***) that is non-vacuous.*

Therefore for every non-vacuous IGR and LIGR there is a unique parameter (q) which is the length of the shortest IRR that can be derived using them. The finite values of this parameter were obtained for the LIGR's in Table **??** by comparing them with the list of IRR's obtained from the computer output and searching with a text editor.

Unfortunately it is conceivable that an IRR matches an LIGR without it being non-vacuous. It could happen if the IGR including some of its context

matches the IRR. Then when it the extended IGR is reduced to shortest form i.e. an IGR then it might no longer contain the LIGR so the LIGR has not been proved to be non-vacuous.

$$\mathtt{3b^5\underline{d}} \rightarrow \mathtt{3\_bababa} \begin{cases} \overset{\mathtt{d}}{\rightarrow} \mathtt{1\_(ab)^3a} \begin{cases} \overset{\mathtt{a}}{\rightarrow} \mathtt{1\_aba|ababa(1\underline{a}ab \rightarrow 1\_aba)} \\ \overset{\mathtt{b}}{\rightarrow} \mathtt{3\_ba|bababa(3b\underline{d} \rightarrow 3\_ba)} \\ \overset{\mathtt{c}}{\rightarrow} \mathtt{3\_bababa|ba(3c\underline{d}baba \rightarrow 3\_bababa)} \end{cases} \\ \overset{\mathtt{c}}{\rightarrow} \mathtt{3\_baba|aba(3\underline{c}bab \rightarrow 3\_baba)} \end{cases} \tag{83}$$

$$\begin{aligned} &\mathtt{1\underline{a}ab \rightarrow 1\_aba} \\ &\mathtt{3b\underline{d} \rightarrow 3\_ba} \\ &\mathtt{3c\underline{d}baba \rightarrow 3\_bababa} \\ &\mathtt{3\underline{c}bab \rightarrow 3\_baba} \end{aligned} \tag{84}$$

For example to do this for the LIGR $\mathtt{3Tbb\underline{b}} \leftarrow \mathtt{3Tadb\underline{d}}$ start by trying to construct the IRR pattern $\mathtt{X}$ on the left of the IGR. It must start from some CS of the form $\mathtt{3Tbb\underline{b}}$. Going forward with the computation 3 steps gives $\mathtt{1\underline{T}aba}$. Making the rightmost symbol in $\mathtt{T} = \mathtt{a}$ i.e. $\mathtt{T} = \mathtt{T'a}$ allows the computation to continue to $\mathtt{1\underline{T'}abaa}$ in 5 steps (which can be repeated because the result matches $\mathtt{1\underline{T}aba}$) while $\mathtt{1T'\underline{b}aba} \rightarrow \mathtt{3\underline{T'}baba} = \mathtt{3T^{(2)}\underline{d}baba} \rightarrow \mathtt{1\underline{T^{(2)}}ababa}$ in two steps (which can also be repeated). Also with $\mathtt{T} = \mathtt{T'c}$ it continues to $\mathtt{3T'bbcb\_}$. This ending at the right is required for $\mathtt{X}$ to be an extendable IRR which is in this case $\mathtt{3T'cbb\underline{b}} \rightarrow\rightarrow \mathtt{3T'bbcb\_}$. Now applying $\mathtt{F}$ with $\alpha = \mathtt{b}$ gives $\mathtt{1cbbb\underline{b}} \rightarrow\rightarrow \mathtt{3b^5\_}$ having length 5. This is the required IRR when the redundant symbol $\mathtt{T'}$ has been removed and is clearly the shortest one possible. The repeatable rules are (when reduced to the shortest forms) $\mathtt{1\underline{a}ab \rightarrow 1\_aba}$ which iterates to $\mathtt{1a^n\underline{a}ab \rightarrow 1\_aba^{n+1}}$ and $\mathtt{1d\underline{b} \rightarrow 1\_ab}$ which iterates to $\mathtt{1\underline{(db)}^n} \rightarrow \mathtt{1\_(ab)^n}$ and $\mathtt{3\underline{c}baba} \overset{9}{\rightarrow} \mathtt{3\_babaa}$ which iterates to $\mathtt{3c^{n-1}\underline{c}baba} \rightarrow \mathtt{3\_babaa^n}$. These results can be represented compactly as follows where the new symbols at the pointer needed are given above the arrows, and optionally the number of TM steps in parentheses and a number in parentheses after a CS denotes the length of the symbol string involved when a match with a previous CS substring has occurred and in the computation the pointer does not go beyond this substring so it can be repeated indefinitely.

$$\mathtt{3bb\underline{b}} \overset{3}{\rightarrow} \mathtt{1\_aba} \begin{cases} \overset{\mathtt{a}(5)}{\rightarrow} \mathtt{1\_aba|a(1\underline{a}ab \rightarrow 1\_aba)} \\ \overset{\mathtt{b}}{\rightarrow} \mathtt{3\_b|aba} \begin{cases} \overset{\mathtt{d}}{\rightarrow} \mathtt{1\_a|baba(1d\underline{b} \rightarrow 1\_ab)} \\ \overset{\mathtt{c}(9)}{\rightarrow} \mathtt{3\_baba|a(3\underline{c}bab \rightarrow 3\_baba)} \end{cases} \\ \overset{\mathtt{c}}{\rightarrow} \mathtt{3bbcb\_(??)} \end{cases} \tag{85}$$

$$\begin{array}{l} 1\underline{a}aab \xrightarrow{5} 1\_aba \\ 1d\underline{b} \to 1\_ab \\ 3\underline{c}bab \to 3\_baba \end{array} . \tag{86}$$

$$1\underline{c}abab \to 3b^5\_ \begin{cases} \xrightarrow{d} 3\_bababa() \\ \xrightarrow{c} 3b^5|c\_(3\underline{c} \to 3c\_) \end{cases} \tag{87}$$

$$\begin{array}{l} 3c\underline{d}ac \to 3bbbc\_ \\ 3\underline{c}dbad \to 3b^5\_ \\ 3\underline{c} \to 3c\_ \\ 2\underline{b} \to 2c\_ \\ 2\underline{b}ca \to 2bbb\_ \\ 1\underline{c} \to 1b\_ \\ 3\underline{c} \to 3c\_ \end{array} \tag{88}$$

By putting the extra symbol c on the right, most of the results can just be copied from above, but the condition for the repetition works out slightly differently because the first CS $3b^5\_$ does not appear.

$$1\underline{c}ababc \xrightarrow{c} 3b^5c\_ \begin{cases} \xrightarrow{d} 2b^5|bb\_ \begin{cases} \xrightarrow{a} 3b^7|b\_ \begin{cases} \xrightarrow{d} 1\_a(ba)^4(??) \\ \xrightarrow{c} 3b^5bbb|c\_ \begin{pmatrix} 3\underline{c} \to 3c\_, \\ 3c\underline{d}ac \to 3bbbc\_ \end{pmatrix} \end{cases} \\ \xrightarrow{b} 2b^7|c\_(2\underline{b} \to 2c\_) \\ \xrightarrow{c} 3\_(ba)^3bc(??) \end{cases} \\ \xrightarrow{c} 3b^5c|c\_(3\underline{c} \to 3c\_) \end{cases} \tag{89}$$

$$1\underline{c}abc \to 1bbbb\_ \begin{cases} \xrightarrow{a} 2b^5\_ \begin{cases} \xrightarrow{a} 3b^6\_ \begin{cases} \xrightarrow{d} 1\_a(ba)^3(??) \\ \xrightarrow{c} 3b^6|c\_(3\underline{c} \to 3c\_) \end{cases} \\ \xrightarrow{b} 2b^5|c\_(2\underline{b} \to 2c\_) \\ \xrightarrow{c} 3\_bababc(??) \end{cases} \\ \xrightarrow{b} 3\_babab(??) \\ \xrightarrow{c} 1b^4|b\_(1\underline{c} \to 1b\_) \end{cases} \tag{90}$$

$$1\underline{c}aa \to 3bbb\_ \begin{cases} \xrightarrow{d} 3\_baba(??) \\ \xrightarrow{c} 3bbb|c\_(3\underline{c} \to 3c\_) \end{cases} \tag{91}$$

$$1\underline{c}a \to 2bb\_ \begin{cases} \xrightarrow{a} 3bbb\_ \begin{cases} \xrightarrow{d} 3\_baba(??) \\ \xrightarrow{c} 3bbb|c\_(3\underline{c} \to 3c\_) \end{cases} \\ \xrightarrow{b} 2bb|c\_(2\underline{b} \to 2c\_) \\ \xrightarrow{c} 1\_abc(??) \end{cases} \tag{92}$$

$$\begin{array}{l} 2\underline{a}cd \to 2bbb\_ \\ 2\underline{b}ca \to 2bbb\_ \end{array} \tag{93}$$

$$1\underline{c}c \to 1bb\_ \begin{cases} \xrightarrow{a} 2bb|b\_ \begin{cases} \xrightarrow{a} 3b^3|b\_ \begin{cases} \xrightarrow{d} 1\_ababa(??) \\ \xrightarrow{c} 3b^4|c\_(3\underline{c} \to 3c\_) \end{cases} \\ \xrightarrow{b} 2bbb|c\_(2\underline{b} \to 2c\_) \\ \xrightarrow{c} 3\_babc(??) \end{cases} \\ \xrightarrow{b} 3\_bab(??) \\ \xrightarrow{c} 1bbb\_(1\underline{c} \to 1b\_) \end{cases} \tag{94}$$

This method can be applied starting with each state and symbol pair to generate the following results in the form of trees. The hope was to be able to join all these up so that where F needs to be invoked the logic can pass to another tree, but there is problem with "carries" i.e. strings of symbols or context that needs to be carried to the next tree and added for the computation to be followed. Only when the TM is going effectively in a single direction because of a repeating condition can these be ignored. Clearly this method can, if all the repeating cycles are picked out, produce all of these starting from the simplest single step and single symbol cases to the more complex case in equation (**??**), but it cannot make sense of a big cycle, if there is one, without somehow abstracting away the effect of these "carries".

The effect of "carries" is to cycle round within a single tree because the trees are designed to be closed under the effect of extra symbols added unless it results in a change of direction.

A single tree gives a recursive definition of all the IRR's that can be obtained from the given LHS of the LIGR that motivated their construction and so allows a proof of the existence or non-existence of IRR's generated by the LIGR.

In the connected diagram write just the endpoints after the change of direction.

A branch can end because of three conditions, (1) a repetition or loop (indicating that there is no point in continuing) because a substring in the same branch has been developed before and (2) a reference to a loop. This is indicated by the loop label (a greek letter) and an asterisk and (3) when the computation from the same CS appears on another branch. If the computation ends in a subset of a CS previously developed, the extra symbol(s) need to be added resulting in another subtree. A very simple case follows. To get the results for 3b|b_ from those for 3|b_ ($\epsilon$ an extra b must be put on the left. The first branch gives rise to the loop $\delta$ going left and the added b gives $3\underline{b}ba \to 1\_aba$ and in its new location it gives rise to a new loop $1\underline{a}ad \to 1\_aba$. The branch to 3b|c_ going to the loop $\theta$ going to the right, gives this same loop after the b is added on the left because the extra symbol is added on the left.

If the new symbol is added on the opposite side to the direction of travel in a cycle of the loop the same loop with be obtained after the symbol is added, but not otherwise.

How to handle reversals of direction add the extra symbol to extend the tree as above or start a new one? a reversal of direction so that the pointer is on the other side of the string. In this case the computation continues on another tree because every possible state and symbol pair (accounting for all possible cases) is at the root of a tree

Another example is how the development of $\mathtt{2|bb\_}$ is obtained from that of $\mathtt{2|b\_}$. Putting a $\mathtt{b}$ on the left gives the first result $\mathtt{1\underline{b}aba} \rightarrow \mathtt{3\_baba}$. This results in two loops because both starting points $\mathtt{3\underline{c}bad}$ and $\mathtt{3bbb\underline{d}}$ match the endpoint $\mathtt{3\_baba}$. This is indicated by $\mathtt{3\underline{c}bad} \rightarrow \mathtt{3bbb\underline{d}} \rightarrow \mathtt{3\_baba}$. The other cases are very easy. Note that the | has no meaning unless each symbol is added one at a time so they are omitted if this does not happen.

In case (1) in what follows the numbers in typewriter font represent different repeating conditions. The number is the length $l$ of the string over which a repetition can occur. This includes the symbol added so it is always $\geq 1$ and if $\mathtt{1} = \mathtt{1}$ there is no symbol string to be matched. The repeating condition is where the state, pointer position (right or left) and the symbol string match between a CS and another CS that is in the path to it from the root of the tree. Between these two CS's the pointer moves in a range. A repetition also requires every symbol in the second CS in the range to match the corresponding string from the first CS.

In case (2) a branch ends because the computation (taken as far as possible) goes in the opposite direction. This is indicated by an italicised identification number. These numbers are also repeated where a matching CS appears in another tree from where the computation can continue.

$$
\mathtt{1\_}\begin{cases}
\xrightarrow{\mathtt{a}} \mathtt{2|b\_}\begin{cases}
\xrightarrow{\mathtt{a}} \mathtt{3b|b\_}(\epsilon)\begin{cases}
\xrightarrow{\mathtt{d}} \mathtt{1\_aba}(\mathtt{1\underline{a}ad} \rightarrow \mathtt{1\_aba}) \\
\xrightarrow{\mathtt{c}} \mathtt{3bb|c\_}(\theta^*)
\end{cases} \\
\xrightarrow{\mathtt{b}} \mathtt{2b|c\_}(\zeta^*) \\
\xrightarrow{\mathtt{c}} \mathtt{3\_bc|}(\beta^*)
\end{cases} \\
\xrightarrow{\mathtt{b}} \mathtt{3\_b|}\begin{cases}
\xrightarrow{\mathtt{d}} \mathtt{1\_a|b}(\mathtt{1d\underline{b}} \rightarrow \mathtt{1\_ab}) \\
\xrightarrow{\mathtt{c}} \mathtt{2|bb\_}(\alpha)\begin{cases}
\xrightarrow{\mathtt{ad}} \mathtt{3\_b|aba}(\mathtt{3\underline{c}bad} \rightarrow \mathtt{3bbb\underline{d}} \rightarrow \mathtt{3\_baba}) \\
\xrightarrow{\mathtt{ac}} \mathtt{3bbb|c\_}(\theta^*) \\
\xrightarrow{\mathtt{b}} \mathtt{2bb|c\_}(\zeta^*) \\
\xrightarrow{\mathtt{c}} \mathtt{1\_abc|}(\eta^*, \text{ignore x})
\end{cases}
\end{cases} \\
\xrightarrow{\mathtt{c}} \mathtt{1|b\_}(\mathtt{1\underline{c}} \rightarrow \mathtt{1b\_}, \gamma)
\end{cases}
\tag{95}
$$

$$2\_\begin{cases} \xrightarrow{a} 3|b\_(\epsilon)\begin{cases} \xrightarrow{d} 3\_ba|(\delta^*) \\ \xrightarrow{c} 3b|c\_(\theta^*) \end{cases} \\ \xrightarrow{b} 2|c\_(2\underline{b} \to 2c\_, \zeta) \\ \xrightarrow{c} 1\_c|\begin{cases} \xrightarrow{d} 3\_bc|(\beta) \overset{x=a,b,c}{\to} 1\_abc(\eta)\begin{cases} \xrightarrow{a} 1\_aba|c(1ab\underline{b} \to 1\underline{a}ab \to 1b\underline{b}a \to 1\_aba) \\ \xrightarrow{b} 3\_b|abc\begin{cases} (x = d, 3b\underline{d} \to 3\_ba) \\ \overset{d,x=c}{\to} 1\_a|babc(1d\underline{b} \to 1\_ab) \\ \overset{c,x=c}{\to} 3\_baba|c(3\underline{c}bab \to 3\_baba) \end{cases} \\ \xrightarrow{c} 1b^4\_(\gamma^*) \end{cases} \\ \xrightarrow{c} 1bb\_(\gamma^*) \end{cases} \end{cases} \tag{96}$$

$$3\_\begin{cases} \xrightarrow{d} 1\_a|\begin{cases} \xrightarrow{a} 3|bb\_(\epsilon^*) \\ \xrightarrow{b} 3\_b|a(3b\underline{d} \to 3\_ba, \delta) \\ \xrightarrow{c} 2|bb\_(\alpha^*) \end{cases} \\ \xrightarrow{c} 3|c\_(3\underline{c} \to 3c\_, \theta) \end{cases} \tag{97}$$

Do the results (??),(??) and (??) adequately characterise TM (??)?

By taking the longest results of the repeating cycles on the RHS's of (??),(??) and (??) i.e. 1_aba, 3_baba and adding every symbol in turn gives $1\underline{a}aba \to 1\_abaa, 1\underline{a}abaa \to C, 1\underline{b}abaa \to E, 1\underline{c}abaa \to 3b^5\_, 1\underline{b}aba \to 3\_baba, 3\underline{a}baba \to A, 3\underline{b}baba \to A, 3\underline{c}baba \to E, 1\underline{c}aba \to 2bbbb\_$ where I am using the capital letter notation for pseudo-states in (??). Another round of this will generate all the results of (??).

$$1\underline{a} \to 2b\_\begin{cases} \xrightarrow{a} 3b|b\_\begin{cases} \xrightarrow{d} 1\_aba(??) \\ \xrightarrow{c} 3bb|c\_(3\underline{c} \to 3c\_) \end{cases} \\ \xrightarrow{b} 2b|c\_(2\underline{b} \to 2c\_) \\ \xrightarrow{c} 3\_bc(??) \end{cases} \tag{98}$$

$$1\underline{b} \to 3\_b\begin{cases} \xrightarrow{d} 1\_a|b\begin{cases} \xrightarrow{a} 1\_aba|(1\underline{a}ab \to 1\_aba) \\ \xrightarrow{b} 3\_b|ab(3b\underline{d} \to 3\_ba) \\ \xrightarrow{c} 2bbc\_(??) \end{cases} \\ \xrightarrow{c} 2|bb\_(??) \end{cases} \tag{99}$$

$$1\underline{c} \rightarrow 1b\_ \begin{cases} \xrightarrow{a} 2b|b\_ \begin{cases} \xrightarrow{a} 3bb|b\_ \begin{cases} \xrightarrow{d} 3\_baba|(??) \\ \xrightarrow{c} 3bbb|c\_(3\underline{c} \rightarrow 3c\_) \end{cases} \\ \xrightarrow{b} 2bb|c\_(2\underline{b} \rightarrow 2c\_) \\ \xrightarrow{c} 1\_abc(??) \end{cases} \\ \xrightarrow{b} 1\_ab(??) \\ \xrightarrow{c} 1b|b\_(1\underline{c} \rightarrow 1b\_) \end{cases} \tag{100}$$

$$2\underline{a} \rightarrow 3b\_ \begin{cases} \xrightarrow{d} 3\_ba(??) \\ \xrightarrow{c} 3b|c\_(3\underline{c} \rightarrow 3c\_) \end{cases} \tag{101}$$

$$2\underline{b} \rightarrow 2c\_ \begin{cases} \xrightarrow{a} 3c|b\_ \begin{cases} \xrightarrow{d} 3|bbb\_ \begin{cases} \xrightarrow{d} 3\_baba(??) \\ \xrightarrow{c} 3bbb|c\_(3\underline{c} \rightarrow 3c\_) \end{cases} \\ \xrightarrow{c} 3cb|c\_(3\underline{c} \rightarrow 3c\_) \end{cases} \\ \xrightarrow{b} 2c|c\_(2\underline{b} \rightarrow 2c\_) \\ \xrightarrow{c} 1|bb\_ \begin{cases} \xrightarrow{a} 2bb|b\_ \begin{cases} \xrightarrow{a} 3b^3|b\_ \begin{cases} \xrightarrow{d} 1\_ababa(??) \\ \xrightarrow{c} 3b^4|c\_(3\underline{c} \rightarrow 3c\_) \end{cases} \\ \xrightarrow{b} 2bbb|c\_(2\underline{b} \rightarrow 2c\_) \\ \xrightarrow{c} 3\_babc(??) \end{cases} \\ \xrightarrow{b} 3\_bab(??) \\ \xrightarrow{c} 1bb|b\_(1\underline{c} \rightarrow 1b\_) \end{cases} \end{cases} \tag{102}$$

$$
2\underline{c} \to 1\_c \begin{cases}
\xrightarrow{a} 3\_bc| \begin{cases}
\xrightarrow{a} 1\_a|bc \begin{cases}
\xrightarrow{a} 1\_aba|c(1\underline{a}ab \to 1\_aba) \\
\xrightarrow{b} 3\_b|abc(3b\underline{a} \to 3\_ba) \\
\xrightarrow{c} 1b^4\_(??)
\end{cases} \\
\xrightarrow{b} 1\_a|bc \begin{cases}
\xrightarrow{a} 1\_aba|c(1\underline{a}ab \to 1\_aba) \\
\xrightarrow{b} 3\_b|abc(3b\underline{b} \to 3\_ba) \\
\xrightarrow{c} 1b^4\_(??)
\end{cases} \\
\xrightarrow{c} 1\_abc| \begin{cases}
\xrightarrow{a} 1\_aba|c(1\underline{a}ab \to 1\_aba) \\
\xrightarrow{b} 3\_b|abc(\text{no rep but treat as above}) \\
\xrightarrow{c} 1b^4\_(??)
\end{cases}
\end{cases} \\
\xrightarrow{b} 3\_b|c \begin{cases}
\xrightarrow{a} 1\_a|bc \begin{cases}
\xrightarrow{a} 1\_aba|c(1\underline{a}ab \to 1\_aba) \\
\xrightarrow{b} 3\_b|abc(3b\underline{a} \to 3\_ba) \\
\xrightarrow{c} 1b^4\_(??)
\end{cases} \\
\xrightarrow{b} 1\_a|bc \begin{cases}
\xrightarrow{a} 1\_aba|c(1\underline{a}ab \to 1\_aba) \\
\xrightarrow{b} 3\_b|abc(3b\underline{b} \to 3\_ba) \\
\xrightarrow{c} 1b^4\_(??)
\end{cases} \\
\xrightarrow{c} 1\_abc| \begin{cases}
\xrightarrow{a} 1\_aba|c(1\underline{a}ab \to 1\_aba) \\
\xrightarrow{b} 3\_b|abc(\text{no rep but treat as above}) \\
\xrightarrow{c} 1b^4\_(??)
\end{cases}
\end{cases} \\
\xrightarrow{c} 1bb\_(??)
\end{cases}
\tag{103}
$$

This can apparently be continued indefinitely and is equivalent to Table **??**. In this case it arises just from the method illustrated starting on the top of page 59. When this has been checked carefully probably this paper can be drastically shortened.

$$
3\underline{d} \to 1\_a \begin{cases}
\xrightarrow{a} 3bb\_(??) \\
\xrightarrow{b} 3\_b|a \begin{cases}
\xrightarrow{d} 1\_a|ba(1d\underline{b} \to 1\_ab) \\
\xrightarrow{c} 3bbb\_(??)
\end{cases} \\
\xrightarrow{c} 2bb\_(??)
\end{cases}
\tag{104}
$$

$$
3\underline{c} \to 3c\_ \begin{cases}
\xrightarrow{d} 2|bb\_ \begin{cases}
\xrightarrow{a} 3bb|b\_ \begin{cases}
\xrightarrow{d} 3\_baba(??) \\
\xrightarrow{c} 3bbb|c\_(3\underline{c} \to 3c\_)
\end{cases} \\
\xrightarrow{b} 2bb|c\_(2\underline{b} \to 2c\_) \\
\xrightarrow{c} 1\_abc(??)
\end{cases} \\
\xrightarrow{c} 3c|c\_(3\underline{c} \to 3c\_)
\end{cases}
\tag{105}
$$

It may also be interesting to look at the relationships between the repetitions as the result of the computation. The distinct repetitions found in the

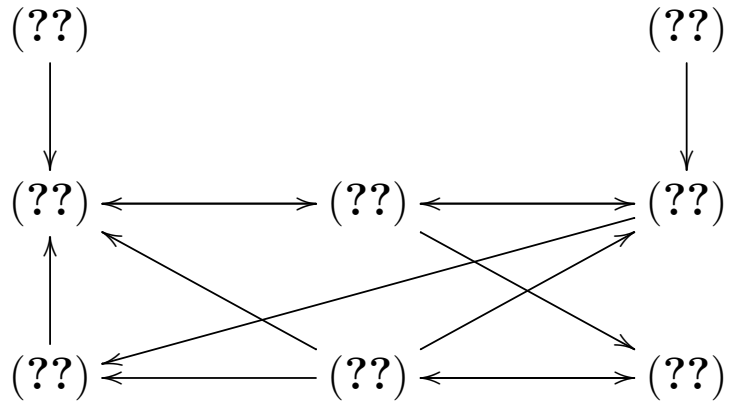


Figure 3: relationships between the 'trees'

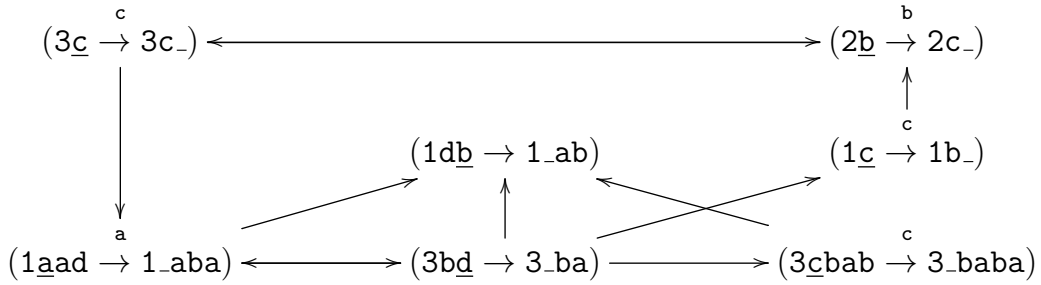


Figure 4: A schematic summary of the results in the 'trees'. The parentheses are loops that repeat if certain symbol combinations repeat. This is an attempt to describe the complete algorithm of TM (**??**). There are other connections from the first part to the second part which seems to coincide with Table **??**. The labels (next symbol seen) on the arrows are above the arrows if viewed such that the arrow goes to the right.

results (**??**) to (**??**) are

$$\begin{array}{ll} (1) & 3\underline{c} \rightarrow 3c\_ \\ (2) & 2\underline{b} \rightarrow 2c\_ \\ (3) & 1\underline{a}ab \rightarrow 1\_aba \\ (4) & 3b\underline{d} \rightarrow 3\_ba \\ (5) & 1\underline{c} \rightarrow 1b\_ \\ (6) & 1d\underline{b} \rightarrow 1\_ab \end{array} \tag{106}$$

The results in the 'trees' can be perhaps found more simply by starting just with each state (instead of state and symbol pairs) and matching going either way.

# 8 Formulating the condition for a repetition

In these trees if on any branch, the final CS matches an earlier CS in such a way that the loop can be repeated (this might work after some symbols not involved in the loop itself are ignored) then the algorithm terminates the branch because continuing is a special case of what has already been done. If this happens, all the CS's that match the final CS should be listed in order in parentheses, so that many other results of the TM can be found easily. Suppose

$$\texttt{CS}_1 \overset{\alpha_1}{\to} \texttt{CS}_2 \overset{\alpha_2}{\to} \ldots \tag{107}$$

is such a branch that ends in a repeating loop and the pointer is at the right in each CS where $\texttt{CS}_i$ has length $i$ and in step $i$ from $\texttt{CS}_i$ to $\texttt{CS}_{i+1}$ the pointer reaches and uses $r(i)$ symbols (this excludes the last symbol arrived at that is not yet read). $\texttt{CS}_1$ is a CS of length 1 with the pointer on the right and is just one TM step away from the root which is a state and symbol pair. After $l_2 - 1$ steps in (**??**) giving a CS of length $i = l_2$ what is the condition for a repetition of an earlier CS of length $i = l_1$? This involves $l_2 - l_1$ steps in (**??**).

Because the symbols are added on the right, the tape positions will be counted going to the right and the leftmost position is position 1 in all the CS's. The pointer starts at $l_1 + 1$ and first goes to $l_1 + 2$ via $l_1 - r(l_1) + 2$ (the symbol positions used go from $l_1 - r(l_1) + 2$ to $l_1 + 1$ i.e. a segment of length $r(l_1)$). The complete potentially repeating computation reaches the following extreme pointer positions in this order $l_1 + 1, l_1 + 2 - r(l_1), l_1 + 2, l_1 + 3 - r(l_1 + 1), \ldots l_2$ because in the final step to get $\texttt{CS}_{l_2}$ the pointer does not go beyond $l_2$. The range of the tape affected by the computation is from position $p$ to $l_2$ inclusive (see Figure **??**) where

$$p = \min_{l_1 \le i \le l_2 - 1} \{i + 2 - r(i)\} . \tag{108}$$

The repeating condition implies that the states match between the start and end of the computation and there is a pair of matching substrings of $m$ symbols in the two CS's such that each substring lies within the range $p$ to $l_2$ and must include all the symbols in that range on the left hand end otherwise the computation could not be repeated due to a mismatch. Therefore $p = l_1 - m + 1$ is the leftmost symbol position involved in the matching i.e. $m = l_1 - p + 1$. The length of the potentially repeating rule is the length of tape involved in (**??**) i.e. $t = l_2 - p + 1$. Therefore $t - m = l_2 - l_1 \ge 1$. One of the shortest possible examples is $\texttt{3c} \to \texttt{3c\_}$. If there are no other symbols on the left, $l_1 = 0$ and $l_2 = 1$ and $p = 1$ therefore $m = 0$ and $t = 1$ so in general $t > m \ge 0$.

In the results (**??**) to (**??**) the notation | was introduced in the CS's to indicate the limit beyond which the pointer did not go to obtain the CS from the preceding one. This is a visual indication of $r(i)$ which is the number of symbols between | and the end of the string where the symbol _ is, where $i$ is the length of the *preceding* CS.

| $i+2-r(i)$ | $r(i)$ | $i$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 2 | 4 | x | x | x | x | _ | | | |
| 3 | 4 | 5 | x | x | x | \| x | x | _ | | |
| 5 | 3 | 6 | x | x | \| x | x | x | x | _ | |
| 8 | 1 | 7 | x | x | x | x | \| x | x | x | _ |
| minimum: $p=3$ | | 8 | x | x | x | x | x | x | x | \| x | _ |
| | | | | | ↑ | | | | | | |

Figure 5: **A schematic example of a repetition (states omitted)**. Here $l_1 = 4$,$l_2 = 8$, and $p = 3$ therefore $m = 2$ and $t = 6$ and the repeating rule has the form $\widehat{\texttt{xx}}\underline{\texttt{x}}\texttt{xxx} \rightarrow \texttt{xxxx}\widehat{\texttt{xx}}\texttt{\_}$ where the x's represent any symbols, and the _'s are where the symbols are added at the pointer position. The strings of symbols under the widehat $\widehat{\ }$ must be the same. These are the $m$ symbols that are repeated. The ↑ is where $p = 3$ giving a visual indication of the end of range of the symbols that are involved in the repeating computation rule.

**********

The different end points are also related by a graph expressing the relation "is followed by" which is as follows:
$1 \rightarrow 2, 2 \rightarrow \{3, 7, 8, 10\}, 3 \rightarrow \{7, 10\}, 4 \rightarrow \{8, 10, 11\}, 5 \rightarrow \{4, 10, 11\}, 6 \rightarrow \{3, 4\}$
$, 7 \rightarrow 3, 8 \rightarrow 10, 9 \rightarrow 1, 11 \rightarrow \{3, 8\}, 12 \rightarrow 7$
An appropriate way to understand what this tells us about the computation is probably to first deal with the strongly connected subgraph consisting of *3*, *7* then add to it other related repeating conditions such as *8,10* etc.

The endpoints of the trees that are not endpoints of cycles are the result of a change of direction of the TM and the repeated action of a cycle of the TM indicated by an endpoint that is a repeating condition. The connection between these endpoints and cycles needs to be made manifest.

[ There is however a limitation to completely ignoring the RHS's which is that IGR's could be generated (vacuous IGR's) which can never be used in practice. To ensure this does not happen i.e. that W in non-vacuous, the LHS of the IGR W must be an IRR pattern that matches an IRR of extendable type say X because then W can be applied to X giving another IRR Y, (which of course assumes that the logic of W is correct). Therefore such an LIGR $\texttt{W}^*$, the LIGR associated with IGR W, will also be called non-vacuous. This condition will be followed up later and reduces the set of LIGR's generated by Algorithm **??** to those that are non-vacuous, and will be applied later in section **??**.]

Returning to (**??**), and proceeding with the shortest of these by going forward with the computation to try to generate an extendable IRR gives $\texttt{3b}^3\underline{\texttt{b}} \rightarrow\rightarrow \texttt{3\_baba}$ showing that this is not possible because the pointer ends up at the left showing that the IRR generated has type RLL and so is non-

extendable, so an extendable IRR involving this needs at least one more symbol on the left. By adding each possible symbol on the left, and recording the rightmost position of the pointer ($r$) in each result gives $\mathtt{3\underline{d}baba} \to \mathtt{1\_ababa}$ ($r = 1$) and $\mathtt{3\underline{c}baba} \xrightarrow{9} \mathtt{3\_babaa}$ ($r = 4$) showing that this applies again. One more time gives $\mathtt{1\underline{a}ababa} \to \mathtt{1\_abaaba}$ ($r = 3$), $\mathtt{1\underline{b}ababa} \to \mathtt{3\_bababa}$ ($r = 1$), $\mathtt{1\underline{c}ababa} \to \mathtt{3b^5\_}$ ($r = 6$). This time each of the results gives rise to a final CS `X` which is a subset (having extra symbols) of a CS `Y` such that in computation leading to the `X` there is another CS that is also a subset of `Y` where the pointer does not go beyond the symbols in `Y` to carry out the final computation. This is the condition (`C`) that there is no point in continuing the forward computation any further because once the computation from `Y` to `X` has been found (expressed in its shortest form) it can be applied again, and so on indefinitely. In the first example $\mathtt{1\_ababa} \to \mathtt{1\_abaaba}$ we have $\mathtt{Y} = \mathtt{1\_aba}$. It is often but not always the case that `X` is a subset of `Y`, which condition guarantees that condition `C` holds, but it does not hold in general as in this example. Once condition `C` has been found, the computation terminates and the length of the minimal string over which the computation happens is written in parentheses. If however the original computation continues to the opposite end of the string so that an extendable IRR results, the result of `F` applied to this can be shown.

$$\mathtt{3b^3\underline{b}} \to \mathtt{3\_baba} \begin{cases} \xrightarrow{\mathtt{d}} \mathtt{1\_ababa} \begin{cases} \xrightarrow{\mathtt{a}} \mathtt{1\_abaaba}(3) \\ \xrightarrow{\mathtt{b}} \mathtt{3\_bababa}(2) \\ \xrightarrow{\mathtt{c}} \mathtt{3\_bababa}(6) \end{cases} \\ \xrightarrow{\mathtt{c}} \mathtt{3\_babaa}(4) \end{cases} \tag{109}$$

Doing this again gives nothing new when shortened to the shortest forms which are, (with the number of TM steps above the arrow) as follows with the maximum value of $r = 6$.

$$\begin{array}{l} \mathtt{1\underline{a}ab} \xrightarrow{5} \mathtt{1\_aba} \\ \mathtt{1\underline{d}b} \to \mathtt{1\_ab} \\ \mathtt{3c\underline{d}baba} \to \mathtt{3\_(ba)^3} \\ \mathtt{3\underline{c}bab} \xrightarrow{9} \mathtt{3\_baba} \end{array} . \tag{110}$$

Because of this closure, the these results can be written in Table **??** in which the 5 CS's at the head of the columns behave like pseudo states. This maximum value of $r$ indicates that the maximum number of TM steps to the right the pointer goes to complete one of these cycles is one less i.e. 5 because it starts at position 1 i.e. the pseudo states have length 5. Four of these (A,B,D and E) were obtained from the either side of the results in (**??**) and C came by carrying out the next cycle in detail and just indicated in words in this paragraph. The sufficiency of the set of 5 (coincidentally) pseudo states is confirmed by the closure indicated in Table (**??**). The body of the

table indicates the state transitions as a result of each new symbol read, and the steps all involve moving left by one space so it is effectively a finite state machine. Also note that starting with the other CS's in (**??**) gives the same results because $\mathtt{3b}^5\underline{\mathtt{d}} \to \mathtt{3\_bababa}$ and the first stage of the calculation above gives the same results when expressed in the shortest form.

Table 9: A finite state machine going left derived from TM **??**

| A: 1_ababa | B: 1_abaab | C: 1_abaaa | D: 3_babab | E: 3_babaa |
|---|---|---|---|---|
| a → B | a → C | a → C | a → A | a → A |
| b → D | b → E | b → E | b → A | b → A |
| c → D | c → D | c → D | c → D | c → E |

This argument shows that the IGR's sought do not exist, therefore the LIGR's starting with the LHS's in (**??**) are vacuous and should not be listed. As a result of this, the set of LIGR's derived by hand according to the method above in Section **??** with this modification, and the computer results in Table **??** based on IGR's now agree.

This also shows that the entry into any of these 5 CS's, the pseudo states, is the criterion by which the TM is certain to remain in this finite state machine behaviour indefinitely. If this has not yet been reached, it might be possible that this type of behaviour could be avoided and it can happen indefinitely with some of the results below being examples of this where the TM moves to the right. Note the similarity of this with the results of the next section (Section **??**) derived by a different method.

According to Figure **??** this procedure for checking the LIGR's for being vacuous could have been done after the first stage of analysis of LIGR's 1 and 2 with a single application of F because instead of waiting till they were all found, saving much time, only one of LIGR's $15-19$, one of $24-25$, and one of $26-27$ needs to have been found.

It is easy to show other cases where an unending iteration can occur by applying a very similar method to the other LHS's of LIGR's in (**??**). The results are as follows, each for all $n \geq 0$:

$$\begin{array}{l} \mathtt{1}\underline{\mathtt{c}}\mathtt{c}^n \to \mathtt{1b}^{n+1}\_ \\ \mathtt{1b}^{2n}\underline{\mathtt{b}} \to \mathtt{3\_b(ab)}^n \\ \mathtt{3}\underline{\mathtt{c}}\mathtt{c}^n \to \mathtt{3c}^{n+1}\_ \\ \mathtt{3b}^{2n+1}\underline{\mathtt{b}} \to \mathtt{3\_(ba)}^{n+1} \\ \mathtt{2}\underline{\mathtt{b}}\mathtt{b}^n \to \mathtt{2c}^{n+1}\_ \end{array} . \tag{111}$$

Also, the results (**??**).1, 6 can be similarly iterated to give

$$\begin{array}{l} \mathtt{1a}^n\underline{\mathtt{a}}\mathtt{ab} \to \mathtt{1\_aba}^{n+1} \\ \mathtt{3c}^{n-1}\underline{\mathtt{c}}\mathtt{bab} \to \mathtt{3\_baba}^{n+1} \end{array} \tag{112}$$

respectively. As might be expected from above, each of the cases in (**??**) gives rise to some CS's that cannot be the LHS of some non-vacuous LIGR but these

are not already in the set of LIGR's found. To show how these relationships can work a very simple example worked out in detail follows. Starting from $\mathtt{1\underline{c}a} \to \mathtt{1b\underline{a}} \to \mathtt{2bb\_}$ it follows that $\mathtt{1\underline{c}a}$ cannot be the LHS of a non-vacuous LIGR but in could possibly happen with extra symbol(s) on the right. With $\mathtt{a}$ it gives $\mathtt{2bb\underline{a}} \to \mathtt{3bbb\_}$ showing that the same holds for $\mathtt{1\underline{c}aa}$ and likewise $\mathtt{2bb\underline{b}} \to \mathtt{2bbc\_}$ showing that the same holds for $\mathtt{1\underline{c}ab}$. The latter case can be reduced to just $\mathtt{2\underline{b}} \to \mathtt{2c\_}$ iterating to give the last member of (**??**), however it is not the case that it is clear that every extension of $\mathtt{1\underline{c}a}$ by a symbol on the right will give other examples of CS's which cannot be LHS's of non-vacuous LIGR's. This did happen in the reasoning leading up to Table **??** because of the closure.

Because Table **??** as well as the iterations in (**??**) describe aspects of the behaviour of TM **??** it was interesting to try to join all these up into a more comprehensive description of TM **??** which can at least in part be done as follows:

$$\mathtt{1\underline{c}} \to \mathtt{1b\_}\begin{cases} \xrightarrow{\mathtt{a}} \mathtt{2bb\_} \\ \xrightarrow{\mathtt{b}} \mathtt{1\_ab} \\ \xrightarrow{\mathtt{c}} \mathtt{1bb\_}\begin{cases} \xrightarrow{\mathtt{a}} \mathtt{2bbb\_} \\ \xrightarrow{\mathtt{b}} \mathtt{3\_bab} \\ \xrightarrow{\mathtt{c}} \mathtt{1bbb\_}\begin{cases} \xrightarrow{\mathtt{a}} \mathtt{2b^4\_} \\ \xrightarrow{\mathtt{b}} \mathtt{1\_abab} \\ \xrightarrow{\mathtt{c}} \mathtt{1b^4\_}\begin{cases} \xrightarrow{\mathtt{a}} \mathtt{2b^5\_} \\ \xrightarrow{\mathtt{b}} \mathtt{3\_babab} \to \text{Table } \textbf{??}.\mathtt{D} \\ \xrightarrow{\mathtt{c}} \mathtt{1b^5\_} \end{cases} \end{cases} \end{cases} \end{cases} \tag{113}$$

$$\mathtt{1b^{2n}\_}\begin{cases} \xrightarrow{\mathtt{a}} \mathtt{2b^{2n+1}\_} \to \ (\textbf{??}) \\ \xrightarrow{\mathtt{b}} \mathtt{3\_b(ab)}^n \in \mathtt{T9.D} \text{ if } n \geq 2 \\ \xrightarrow{\mathtt{c}} \mathtt{1b^{2n+1}\_}\begin{cases} \xrightarrow{\mathtt{a}} \mathtt{2b^{2n+2}\_} \to \ (\textbf{??}) \\ \xrightarrow{\mathtt{b}} (\textbf{??}).2 \\ \xrightarrow{\mathtt{c}} \mathtt{1b^{2n+2}\_} \to \ (\textbf{??}) \end{cases} \end{cases} \tag{114}$$

$$2\mathtt{b}^{2n}\_\begin{cases}\xrightarrow{\mathtt{a}} 3\mathtt{b}^{2n+1}\_\begin{cases}\xrightarrow{\mathtt{d}} 1\mathtt{b}^{2n}\underline{\mathtt{b}}\mathtt{a}\to 3\_\mathtt{b}(\mathtt{ab})^n\mathtt{a}\text{ if }n\geq 2\to \mathtt{T9}\ (\mathbf{??}).2\\ \xrightarrow{\mathtt{c}} 3\mathtt{b}^{2n+1}\mathtt{c}\_\xrightarrow{\mathtt{d}} 2\mathtt{b}^{2n+3}\_\to\ (\mathbf{??})\end{cases}\\ \xrightarrow{\mathtt{b}} 2\mathtt{b}^{2n}\mathtt{c}\_\begin{cases}\xrightarrow{\mathtt{a}} 3\mathtt{b}^{2n}\mathtt{cb}\_\begin{cases}\xrightarrow{\mathtt{d}} 1\_(\mathtt{ab})^{n+1}\in \text{T9.A}\\ \xrightarrow{\mathtt{c}} 3\mathtt{b}^{2n}\mathtt{cbc}\_\begin{cases}\xrightarrow{\mathtt{d}} 2\mathtt{b}^{2n}\mathtt{cbbb}\_\begin{cases}\xrightarrow{\mathtt{a}} 3\mathtt{b}^{2n}\mathtt{cb}^4\_\\ \xrightarrow{\mathtt{b}} 2\mathtt{b}^{2n}\mathtt{cb}^3\mathtt{c}\_\\ \xrightarrow{\mathtt{c}} 3\_(\mathtt{ba})^{n+2}\in \mathtt{T9.E}\end{cases}\\ \xrightarrow{\mathtt{c}} 3\mathtt{b}^{2n}\mathtt{cbcc}\_\end{cases}\end{cases}\\ \xrightarrow{\mathtt{b}} 2\mathtt{b}^{2n}\mathtt{cc}\_\begin{cases}\xrightarrow{\mathtt{a}} 2\mathtt{b}^{2n+2}\mathtt{c}\_\in\ (\mathbf{??})\\ \xrightarrow{\mathtt{b}} 2\mathtt{b}^{2n}\mathtt{cbb}\_\\ \xrightarrow{\mathtt{c}} 3\mathtt{b}^{2n}\mathtt{c}^3\_\end{cases}\\ \xrightarrow{\mathtt{c}} 1\mathtt{b}^{2n+2}\_\to\ (\mathbf{??})\end{cases}\\ \xrightarrow{\mathtt{c}} (\mathbf{??}).2\text{ if }n\geq 3\end{cases}\tag{115}$$

$$3\mathtt{c}^{n+1}\_\begin{cases}\xrightarrow{\mathtt{d}} 2\mathtt{c}^n\mathtt{bb}\_\\ \xrightarrow{\mathtt{c}} 3\mathtt{c}^{n+2}\_\in\ (\mathbf{??})\end{cases}\tag{116}$$

$$2\mathtt{c}^{n+1}\_\begin{cases}\xrightarrow{\mathtt{a}} 3\mathtt{c}^{n+1}\mathtt{b}\_\\ \xrightarrow{\mathtt{b}} 2\mathtt{c}^{n+2}\_\in\ (\mathbf{??})\\ \xrightarrow{\mathtt{c}} 1\mathtt{c}^n\mathtt{bb}\_\end{cases}\tag{117}$$

$$3\mathtt{c}^{n+1}\mathtt{b}\_\begin{cases}\xrightarrow{\mathtt{d}} 3\mathtt{c}^n\mathtt{b}^3\_\begin{cases}\xrightarrow{\mathtt{d}} 3\mathtt{c}^{n-1}\underline{\mathtt{c}}\mathtt{bab}\overset{(\mathbf{??}).2}{\to} 3\_\mathtt{baba}^{n+1}\in\text{Table }\mathbf{??}.\mathtt{E}\text{ if }n\geq 0\\ \xrightarrow{\mathtt{c}} 3\mathtt{c}^n\mathtt{b}^3\mathtt{c}\_\end{cases}\\ \xrightarrow{\mathtt{c}} 3\mathtt{c}^{n+1}\mathtt{bc}\_\end{cases}\tag{118}$$

After spending some time doing this which seemed never ending, I noticed that the results in (**??**) or originally based on (**??**), can mostly be continued either by single TM steps to the left or other members of (**??**) and the exceptions to this would be interesting. For example the computation in (**??**).1 can be continued if another symbol (call it $\alpha$) is given at the pointer giving these results when reduced to shortest form: (**??**).1 gives $1\underline{\mathtt{a}}\mathtt{aab}\to 1\_\mathtt{aba}$ i.e. (**??**).1 again if $\alpha=\mathtt{a}$, a single TM step if $\alpha=\mathtt{b}$, and $1\underline{\mathtt{c}}\mathtt{aba}\to 3\mathtt{bbcb}\_$ if $\alpha=\mathtt{c}$ with the last one being the exception because the pointer goes right. Similarly (**??**).2 gives the exception $3\underline{\mathtt{c}}\mathtt{b}\to 2\mathtt{bb}\_$. (**??**).2 gives just (**??**).6, (**??**).4 (**??**).5 give $1\underline{\mathtt{a}}\mathtt{a}\to 3\mathtt{bb}\_$, and (**??**).6-8 give (**??**).6, thus doing this for all the members

of (**??**) the following set of results:

$$\begin{array}{l} \texttt{1}\underline{\texttt{c}}\texttt{aba} \rightarrow \texttt{3bbcb\_} \\ \texttt{3}\underline{\texttt{c}}\texttt{b} \rightarrow \texttt{2bb\_} \\ \texttt{1}\underline{\texttt{a}}\texttt{a} \rightarrow \texttt{3bb\_} \\ \texttt{1}\underline{\texttt{c}}\texttt{a} \rightarrow \texttt{2bb\_} \end{array} \tag{119}$$

Now try to continue (**??**) in the same way generates the following results (going either way)

$$\begin{array}{l} \texttt{3cb}\underline{\texttt{d}} \rightarrow \texttt{3bbb\_} \\ \texttt{2bb}\underline{\texttt{c}} \rightarrow \texttt{1\_abc} \\ \texttt{3bb}\underline{\texttt{d}} \rightarrow \texttt{1\_aba} \\ \texttt{3bbb}\underline{\texttt{d}} \rightarrow \texttt{3\_baba} \end{array} \tag{120}$$

Now try to continue (**??**) in the same way generates the following

$$\begin{array}{l} \texttt{1}\underline{\texttt{c}}\texttt{abc} \rightarrow \texttt{1b}^{\texttt{4}}\texttt{\_} \\ \texttt{3bbb}\underline{\texttt{d}} \rightarrow \texttt{3\_baba} \end{array} \tag{121}$$

The results (**??**), (**??**),(**??**), (**??**) are a subset of the IRR's (ignoring the origins) most likely to be useful for rapidly continuing (**??**) to (**??**).

## 8.1 Obtaining information about a TM from its IGR's

Table **??** shows the frequencies of the different types of IRR's obtained from the computer program [**?**] applied to TM1. Here I have included the TM table itself regarding the steps to the right as RLR and those to the left as LRL.

Table 10: The frequencies of the different types of IRR's

| length | RLL | RLR | LRL | LRR | total |
|---|---|---|---|---|---|
| 1 | 0 | 5 | 4 | 0 | 9 |
| 2 | 1 | 4 | 4 | 3 | 12 |
| 3 | 2 | 2 | 4 | 2 | 10 |
| 4 | 1 | 2 | 5 | 1 | 9 |
| 5 | 0 | 3 | 0 | 1 | 13 |
| 6 | 3 | 2 | 16 | 1 | 22 |
| 7 | 30 | 0 | 0 | 30 | 0 |
| 8 | 0 | 0 | 56 | 0 | 56 |
| 9 | 0 | 0 | 106 | 0 | 106 |
| 10 | 0 | 0 | 201 | 0 | 201 |

Each IGR in Table **??** is a logical implication from the existence of one IRR to another, each containing the arbitrary strings $\texttt{T}_1$ and $\texttt{T}_2$. For example `IGR55` on the right gives an IRR of type `RLL` if $|\texttt{T}_2| = \texttt{0}$. Its length is $|\texttt{T}_1|+\texttt{4} = |\texttt{T}_2|+\texttt{6}$ but needs completion (i.e. further computation as far as possible) if $|\texttt{T}_2| > \texttt{0}$.

IGR51 gives an IRR of type RLR and requires $|\mathtt{T_1}| + 4 = |\mathtt{T_2}| + 3$ which is its length.

The set of IGR's in Table **??** is divided into two parts because the left hand members either represent an IRR of type LRL or RLR (extendable IRR's). The corresponding right hand members represent IRR's of four types such that each IGR is of one of four types as follows: LRL $\Rightarrow$ LRL, LRL $\Rightarrow$ LRR, RLR $\Rightarrow$ RLR, RLR $\Rightarrow$ RLL. The relation "could be followed by" is true where the right hand member of the first IGR matches the left hand member of the second one for suitable strings $\mathtt{T_1}$ and $\mathtt{T_2}$. Thus in many cases the matching will be conditional and this relation will include all possible matches between the IGR's so that the absence of such a relation definitely rules out any possible match. The relation "could be followed by" implies one IGR can follow the other logically to generate a new IRR's from a given one. For example IGR 41 can be followed by IGR 50 provided the last symbol of $\mathtt{T_2}$ is c. Because of this any such relation needs to be verified for the particular case in question.

[ delete? In the case that the number of IGR's is finite as in this example (**??**), the absence of IRR's of type RLR of length $n$ for all sufficiently large $n$ would follow from the absence of a closed cycle of IGR's of type RLR $\Rightarrow$ RLR that can be repeated indefinitely.]

This relation can be represented by a directed graph (digraph) where the nodes are IGR's and is also separated into two parts because of the two types of extendable IRR's. The existence of any circuits should be established for each part of the digraph separately. The existence of such a circuit implies that (provided it can be iterated indefinitely) the number of IRR's of extendable type generated from the TM is infinite. This implies the existence of IRR's of this type for arbitrary large length ($n$). The other case of the absence of a circuit implies their number is finite at least when the number of IGR's is finite.

The search for such closed cycles can be done following from the relation on IGR's defined by "could be followed by" that is easily obtained either from Table **??** by matching the state and the symbols in the neighbourhood of the pointer and has been programmed.

For the example TM **??**, there were no circuits for the digraph of IRR's of type RLR but there were circuits for IRR's of type LRL showing that arbitrarily long IRR's of type LRL could exist but not for IRR's of type RLR. The two circuits of length 1 involve respectively IGR's 38c1 and 39c1. IGR 38 iterated twice gives $2\underline{\mathtt{T_1}} \rightarrow\!\!\rightarrow \mathtt{3\_babT_2} \overset{\mathtt{c}}{\Rightarrow} 2\underline{\mathtt{b}}\mathtt{T_1} \rightarrow\!\!\rightarrow \mathtt{3\_babaT_2} \overset{\mathtt{c}}{\Rightarrow} 2\underline{\mathtt{b}}\mathtt{bT_1} \rightarrow\!\!\rightarrow \mathtt{3\_babaaT_2}$ and it can be clearly iterated $n$ times giving $2\underline{\mathtt{b}}\mathtt{b}^{n-1}\mathtt{T_1} \rightarrow\!\!\rightarrow \mathtt{3\_baba}^n\mathtt{T_2}$. In order to determine $\mathtt{T_1}$ and $\mathtt{T_2}$, the table of IRR's generated from the TM shows that an example of IGR 38 (IRR of length 5 number 5 found by searching for IRR's of the form of the RHS of IGR38) has context pair (bbcb, a) showing that $\mathtt{T_1}$ = bbcb and $\mathtt{T_2}$ = a corresponding to the IRR $2\underline{\mathtt{b}}\mathtt{bcb} \rightarrow\!\!\rightarrow \mathtt{3\_baba}$. Therefore a result of the iteration of 38 is $2\underline{\mathtt{b}}\mathtt{b}^{n-1}\mathtt{bbcb} \rightarrow\!\!\rightarrow \mathtt{3\_baba}^{n+1}$ for $n \geq 0$. There are many other examples of $(\mathtt{T_1}, \mathtt{T_2})$ which could be used here

such as anything resulting from applying IGR14 because this matches the RHS of 38. Another example of this is $2\underline{b}bcac \rightarrow\rightarrow 3\_babac$ giving the result $2\underline{b}b^{n-1}bbcccb \rightarrow\rightarrow 3\_baba^{n+2}b$ for $n \geq 0$. Similarly IGR 39 iterated $n$ times gives $3\underline{c}c^{n-1}T_1 \rightarrow\rightarrow 3\_baba^nT_2$ and an example is $T_1 = ca^3$ and $T_2 = a$. An example of a circuit of length 2 is $22 \cdot 7$. IGR 22 is $1\underline{T_1} \rightarrow\rightarrow 3\_T_2 \overset{22b}{\Rightarrow} 1\underline{c}T_1 \rightarrow\rightarrow 1\_aT_2$. If $T_1$ begins with c then $22{\cdot}7$ can be written as $1\underline{c}T_1 \rightarrow\rightarrow 3\_T_2 \overset{22b}{\Rightarrow} 1\underline{c}cT_1 \rightarrow\rightarrow 1\_aT_2 \overset{7b}{\Rightarrow} 1\underline{a}bcT_1 \rightarrow\rightarrow 3\_baT_2$. However this clearly does not iterate because $1\underline{a}bc$ does not match $1\underline{c}$. This also happens with $1 \cdot 26$. An example of a circuit of length 2 that does iterate is $1 \cdot 22$ which gives $1\underline{T_1} \rightarrow\rightarrow 1\_T_2 \overset{1b}{\Rightarrow} 1\underline{c}T_1 \rightarrow\rightarrow 3\_bT_2 \overset{22}{\Rightarrow} 1\underline{c}cT_1 \rightarrow\rightarrow 1\_abT_2$. This iterates to get $1\underline{c}^{2n}T_1 \rightarrow\rightarrow 1\_(ab)^nT_2$. An example of the first IRR is $1\underline{c}b \rightarrow\rightarrow 1\_ab$. In this case $T_1 = cb$ and $T_2 = ab$ therefore $1\underline{c}^{2n}cb \rightarrow\rightarrow 1\_(ab)^{n+1}$. In these examples, if the middle element of the IRR, the LHS, was added back, the first part of them indicate a recurring move to the right, so there are examples of this.

Because there are no circuits in the digraph of RLR type IGR's, all the IRR's of type RLR are obtained by starting with the RLR IRR's of length 2 and applying all the IGR's of type $RLR \Rightarrow RLR$ until this can be done no more. These IGR's are just those numbered $40 - 44, 46, 47, 50, 51$. Of these IGR's $43, 44, 46$ do not match any others. The IGR's of type RLR generating the IRR(2) are just 40 with context $(c, b)$, 41 with context $(a, b)$ and 47 with context $(c, )$. Carrying this out gives the results in the diagram following where the counts for the numbers of distinct IRR's for each length are given in the top two rows. The frequencies match the IRR results of the computer program and therefore the computer results for the cycles of the IGR's validate the results concerning the moving window behaviour of the TM guessed from Table **??**.

These two results taken together show that arbitrarily long IRR's of type LRL exist but within them, the pointer can move no more than 6 spaces to the left so for example a sequence of CS's of the form $1 \rightarrow 10 \rightarrow 0$ can exist but not if it is of the form $1 \rightarrow 9 \rightarrow x \rightarrow 10 \rightarrow 0$ with $x \leq 4$ because this has a subsequence of the form $9 \rightarrow x \rightarrow 10$ which is an IRR of type RLR of length $10 - x + 1 \geq 7$. If $9 > x > 4$ as soon as the pointer after reaching 10 reaches 5 then the moving window argument applies. Note that this does not stop the TM moving arbitrarily far to the right eg with one of the iterations above following IRR 6.21 such as when all the symbols to its right are c's.

From this it appears that there are no IRR's of length $\geq 7$ of the type RLR, then all such IRR's have type RLL, LRL, or LRR. If this is generally true, and if the TM reaches position 6 followed by position 1 it cannot subsequently reach position $\geq 7$ because this would require a subsequence of CS's of the form $n \rightarrow 1 \rightarrow n + 1$ with $n = 6$ which would be an IRR by lemma **??** of type RLR of length $n + 1$ contradicting the assumption. The pointer is then constrained to positions $\leq 6$, and if it reaches position 0 then because it has reached position 5 previously, the same argument can be applied showing

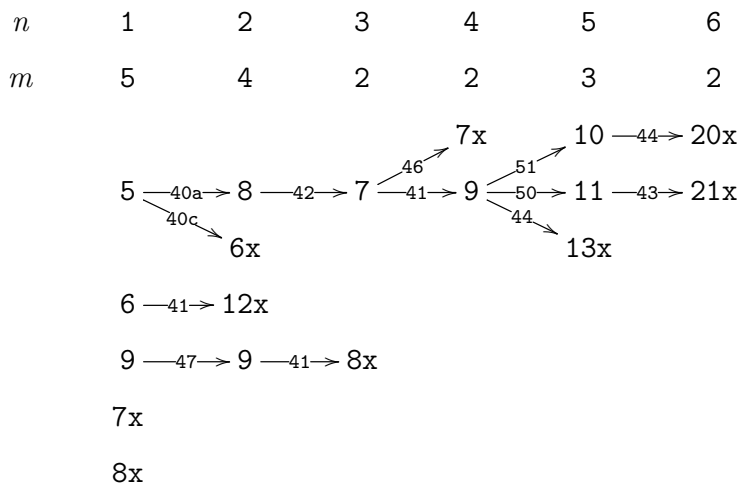


Figure 6: **Provenance of the IRR's of type RLR and lengths up to 6**. An IRR of length 1 is just a single TM step of type being defined as RLR (or LR) if it goes right and LRL (or RL) if it goes left. $n$ is the length of the IRRs, $m$ is the number of IRR's of type RLR (i.e. LR) of length $n$. Only this type of IRR's are shown. Each of the numbers in larger font indicates an IRR of the given length $n$ given in (**??**). The numbers in smaller font are the IGR's in Table **??** needed to generate the IRR on the right of the arrow from the one on its left with letter after it (if present) being $\alpha$. An IRR labelled `x` indicates that no IGR can generate any new IRR of type RLR from it.

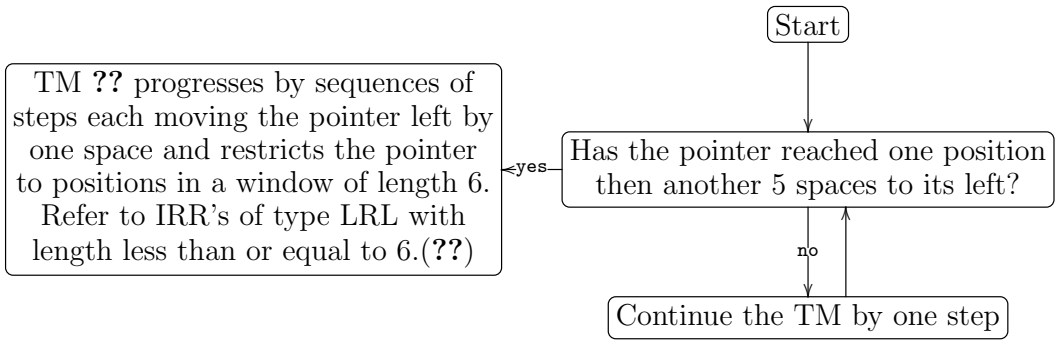


Figure 7: Summary of the results of the analysis of TM(**??**)

that it cannot then reach position $\geq 6$ etc.. This implies that the pointer is constrained to being in a moving window of length 6 that moves left by one space when the pointer moves just to its left. Because of this, if a snapshot is taken of its behaviour whenever the TM reaches just beyond the left hand end of the window, whatever symbol it finds there, the result will be at the next snapshot that the symbols of the window have changed depending on the previous symbols there and the new symbol. Therefore if the TM reaches position 6 followed by position 1 then the above argument involving the moving window applies. This condition of course will happen depending on the initial contents of the tape of the TM that could start the TM doing one of the iterations going right mentioned above.

Now it is obvious that this effective finite state machine is defined by all IRR's of the type LRL of length $\leq n$ for some length $n$. This behaviour going left corresponds to the sequence $\mathtt{A} = \mathtt{p} \to 2 \to \mathtt{p}-1 \to 1 \to \mathtt{p}-2 \to 0$ etc. for some positive integer $\mathtt{p}$ and requires $\mathtt{B}$ (a subsequence of $\mathtt{A}$) i.e. $2 \to \mathtt{p}-1 \to 1$ to exist which is an IRR of type LRL of length $\mathtt{p}$. Also the sequence $\mathtt{p}-2 \to 2 \to \mathtt{p}-1$ would have to exist which is an IRR of type RLR of length $(\mathtt{p}-1)-2+1 = \mathtt{p}-2$ so for this TM $\mathtt{p}$ could not be larger than 8 and the longest IRR of type RLR needed could not have length greater than 6. This is an effective finite state machine with internal state corresponding to the set of symbols in the window and its actual machine state, and it continues indefinitely unless a stationary cycle occurs which would halt it.

In this example the sequence $6 \to 1 \to 7$ is impossible but $6 \to 1 \to 0$ and $6 \to 1 \to 5 \to 0$ are not ruled out.

| | TM table(n = 1) |
|---|---|
| 1 | $3\underline{a} \to 1\_a$ |
| 2 | $3\underline{b} \to 1\_a$ |
| 3 | $2\underline{c} \to 1\_c$ |
| 4 | $1\underline{b} \to 3\_b$ |
| 5 | $1\underline{c} \to 1b\_$ |
| 6 | $1\underline{a} \to 2b\_$ |
| 7 | $2\underline{b} \to 2c\_$ |
| 8 | $2\underline{a} \to 3b\_$ |
| 9 | $3\underline{c} \to 3c\_$ |
| | length = 2 |
| 1 | $1\underline{c}b \to\to 1\_ab$ |
| 2 | $2\underline{a}a \to\to 3\_ba$ |
| 3 | $2\underline{a}b \to\to 3\_ba$ |
| 4 | $2a\underline{c} \to\to 3\_bc$ |
| 5 | $1\underline{a}c \to\to 3\_bc$ |
| 6 | $2c\underline{c} \to\to 1bb\_$ |
| 7 | $2\underline{b}c \to\to 1bb\_$ |
| 8 | $3c\underline{a}$<br>$3c\underline{b} \to\to 2bb\_$ |
| 9 | $1c\underline{b} \to\to 2bb\_$ |
| 10 | $3\underline{c}a \to\to 2bb\_$ |
| 11 | $3\underline{c}b \to\to 2bb\_$ |
| 12 | $3a\underline{a}$<br>$3a\underline{b} \to\to 3bb\_$ |
| | length = 3 |
| 1 | $1aa\underline{b}$<br>$1ab\underline{b} \to\to 1\_aba$ |
| 2 | $1\underline{a}aa \to\to 1\_aba$ |
| 3 | $1\underline{a}ab \to\to 1\_aba$ |
| 4 | $2cb\underline{c} \to\to 1\_abc$ |
| 5 | $1\underline{c}ac \to\to 1\_abc$ |
| 6 | $1\underline{c}cb \to\to 3\_bab$ |
| 7 | $1ca\underline{b}$<br>$1cb\underline{b} \to\to 2bbc\_$ |
| 8 | $3cb\underline{a}$<br>$3cb\underline{b} \to\to 3bbb\_$ |
| 9 | $2\underline{b}aa \to\to 3bbb\_$ |
| 10 | $2\underline{b}ab \to\to 3bbb\_$ |

| | length = 4 |
|---|---|
| 1 | $1\underline{c}ccb$<br>$1\underline{a}bcb \to\to 1\_abab$ |
| 2 | $1cba\underline{b}$<br>$1cbb\underline{b} \to\to 3\_baba$ |
| 3 | $2\underline{b}bcb \to\to 3\_baba$ |
| 4 | $1\underline{c}aaa \to\to 3\_baba$ |
| 5 | $1\underline{c}aab \to\to 3\_baba$ |
| 6 | $1\underline{c}cac$<br>$2\underline{a}cac \to\to 3\_babc$<br>$2\underline{a}cbc$ |
| 7 | $2cab\underline{c}$<br>$2cbb\underline{c} \to\to 1bbbb\_$ |
| 8 | $3\underline{c}cac$<br>$3\underline{c}cbc \to\to 1bbbb\_$ |
| 9 | $3caba$<br>$3cab\underline{b}$<br>$3cbb\underline{a} \to\to 3bbcb\_$<br>$3cbb\underline{b}$ |
| | length = 5 |
| 1 | $1\underline{a}bbcb \to\to 1\_ababa$ |
| 2 | $1\underline{c}caaa$<br>$2\underline{a}caaa$<br>$2\underline{a}cbaa \to\to 1\_ababa$<br>$1\underline{a}baaa$<br>$1\underline{a}baba$ |
| 3 | $1\underline{c}caab$<br>$2\underline{a}cdab \to\to 1\_ababa$<br>$1\underline{a}badb$ |
| 4 | $1\underline{c}ccac$<br>$1\underline{a}dcac \to\to 1\_ababc$<br>$1\underline{a}acbc$ |
| 5 | $2\underline{b}bbcb \to\to 3\_babaa$ |
| 6 | $3\underline{c}cdaa$<br>$2\underline{b}bada \to\to 3\_babaa$ |
| 7 | $3\underline{c}cdab$<br>$2\underline{b}badb \to\to 3\_babaa$ |
| 8 | $1\underline{c}cccb$<br>$1\underline{a}bccb \to\to 3\_babab$<br>$1\underline{c}abcb$ |
| 9 | $2\underline{b}dcac$<br>$2\underline{b}acbc \to\to 3\_babac$ |
| 10 | $3cadb\underline{d} \to\to 3bbbbb\_$ |
| 11 | $1cdbdb \to\to 3bbbbb\_$ |
| 12 | $2\underline{b}bccb \to\to 3bbbbb\_$ |
| 13 | $2cadb\underline{c} \to\to 3bbcbc\_$ |

| | length = 6 |
|---|---|
| 1 | $1\underline{a}bbbcb \to\to 1\_ababaa$ |
| 2 | $2\underline{a}ccdaa$<br>$1\underline{a}bbada \to\to 1\_ababaa$ |
| 3 | $2\underline{a}ccdab$<br>$1\underline{a}bbadb \to\to 1\_ababaa$ |
| 4 | $1\underline{c}c^4b$<br>$1\underline{a}bcccb$<br>$1\underline{c}abccb$<br>$1\underline{c}cabcb \to\to 1\_ababab$<br>$2\underline{a}ccdcb$<br>$2\underline{a}cdbcb$ |
| 5 | $1\underline{a}bdcac$<br>$1\underline{a}bacbc \to\to 1\_ababac$ |
| 6 | $2\underline{b}bbbcb \to\to 3\_babaaa$ |
| 7 | $3\underline{c}ccdaa$<br>$2\underline{b}bbada \to\to 3\_babaaa$ |
| 8 | $3\underline{c}ccdab$<br>$2\underline{b}bbadb \to\to 3\_babaaa$ |
| 9 | $2\underline{b}bcccb$<br>$3\underline{c}cdbcb \to\to 3\_babaab$<br>$3\underline{c}ccdcb$ |
| 10 | $2\underline{b}bdcac$<br>$2\underline{b}bacbc \to\to 3\_babaac$ |
| 11 | $3caadb\underline{d} \to\to 3\_bababa$ |
| 12 | $1cadbdb \to\to 3\_bababa$ |
| 13 | $3cdbdb\underline{d} \to\to 3\_bababa$ |
| 14 | $1\underline{c}abbcb \to\to 3\_bababa$ |
| 15 | $1\underline{a}dcaaa$<br>$1\underline{c}abada$<br>$1\underline{a}acbaa \to\to 3\_bababa$<br>$1\underline{c}ccaaa$ |
| 16 | $2\underline{b}acdaa$<br>$2\underline{b}bcaaa \to\to 3\_bababa$ |
| 17 | $1\underline{a}dcaab$<br>$1\underline{c}abadb$<br>$1\underline{a}acbab \to\to 3\_bababa$<br>$1\underline{c}ccaab$ |
| 18 | $2\underline{b}dcaab$<br>$2\underline{b}acbab \to\to 3\_bababa$ |
| 19 | $1\underline{c}cccac$<br>$1\underline{a}bccac$<br>$1\underline{c}adcac \to\to 3\_bababc$<br>$1\underline{c}aacbc$ |
| 20 | $2caadb\underline{c} \to\to 3b^5c\_$ |
| 21 | $2cdbdb\underline{c} \to\to 3b^5c\_$ |
| 22 | $2\underline{b}bccac \to\to 3b^5c\_$ |

The IRR's of (**??**) includes many examples of different IRR's with the same RHS's and examples where many origins correspond to one RHS. The distinction between these cases is because two IRR's are the same if and only if they have the same middle member (originally known as the LHS) and is omitted for brevity. $\mathtt{40} \to (\mathtt{b},)\mathtt{42} \to (\mathtt{bb},\mathtt{c})\mathtt{41} \to \mathtt{51}$ These specialisations remove the conditions that otherwise would apply to the connections between the IGR's. Can this be done systematically to all the IGR's?

$\mathtt{1}\underline{\mathtt{T_1}}\mathtt{bb} \to\!\to 2\mathtt{T_2c\_}, \mathtt{3T_1}\underline{\mathtt{b}} \to\!\to 2\mathtt{T_2\_}$

Show this gives rise to finite state machine behaviour and relate it to $n_{\mathtt{L}}$ and $n_{\mathtt{R}}$ in my first TM paper.

If the IGR's are related by "can be followed by" it is not necessarily true that if $\mathtt{A} \to \mathtt{B}$ and $\mathtt{B} \to \mathtt{C}$ then $\mathtt{A} \to \mathtt{B} \to \mathtt{C}$. What are the extra conditions? If they could all be found then all the conditions for closed cycles could be found. Find all IGR's $\mathtt{x}$ such that $\mathtt{x} \to \mathtt{x}$. Combining RLR iterations with LRL iterations for a TM with cycles in both cases. This would give rise to possibly repeating cycles that might be describable as a simulated TM.

## 8.2 Approaches to the analysis of a simpler example with quite a different behaviour

Consider another example given by

$$\begin{array}{ll} \mathtt{1}\underline{\mathtt{a}} \to \mathtt{2b\_} & \mathtt{1}\underline{\mathtt{b}} \to \mathtt{2\_a} \\ \mathtt{2}\underline{\mathtt{a}} \to \mathtt{1b\_} & \mathtt{2}\underline{\mathtt{b}} \to \mathtt{2\_a} \\ \mathtt{3}\underline{\mathtt{a}} \to \mathtt{3a\_} & \mathtt{3}\underline{\mathtt{b}} \to \mathtt{1\_b} \end{array} . \tag{123}$$

Table 11: The frequencies of the different types of IRR's

| length($n$) | RLL | RLR | LRL | LRR | total |
|---|---|---|---|---|---|
| 1 | 0 | 3 | 3 | 0 | 6 |
| 2 | 1 | 1 | 3 | 0 | 5 |
| 3 | 1 | 1 | 2 | 1 | 5 |
| 4 | 1 | 1 | 2 | 0 | 4 |
| 5 | 1 | 1 | 2 | 1 | 5 |
| 6 | 1 | 1 | 2 | 1 | 5 |

For this TM the computer program [**?**] gave the results in Table **??** and it appears that the frequencies in the row for $n = \mathtt{5}$ are then repeated indefinitely for larger values of $n$ which was checked to $n$ up to 20. The results for the

IRR's up to length 8 are given in (**??**)

$$
\begin{array}{ll}
 & \texttt{TM table(n = 1)} \\
1 & 1\underline{a} \to 2b\_ \\
2 & 1\underline{b} \to 2\_a \\
3 & 2\underline{a} \to 1b\_ \\
4 & 2\underline{b} \to 2\_a \\
5 & 3\underline{a} \to 3a\_ \\
6 & 3\underline{b} \to 1\_b \\
\hline
 & \texttt{length} = 2 \\
1 & 3a\underline{b} \to 1\underline{a}b \to 2\_aa \\
2 & 2\underline{a}b \to 1b\underline{b} \to 2\_aa \\
3 & 1\underline{a}b \to 2b\underline{b} \to 2\_aa \\
4 & 3\underline{a}b \to 3a\underline{b} \to 2\_aa \\
5 & \left.\begin{matrix}1\\2\end{matrix}\right\} a\underline{b} \to 2\underline{a}a \to 2bb\_ \\
\hline
 & \texttt{length} = 3 \\
1 & 3ab\underline{b} \to 2\underline{a}ab \to 2\_aaa \\
2 & 1\underline{a}ab \to 1bb\underline{b} \to 2\_aaa \\
3 & 2\underline{a}ab \to 2bb\underline{b} \to 2\_aaa \\
4 & \left.\begin{matrix}1\\2\end{matrix}\right\} ab\underline{b} \to 2\underline{a}aa \to 1bbb\_ \\
5 & 3\underline{a}ab \to 3aa\underline{b} \to 1bbb\_ \\
\hline
 & \texttt{length} = 4 \\
1 & 3abb\underline{b} \to 2\underline{a}aab \to 2\_aaaa \\
2 & 2\underline{a}aab \to 1bbb\underline{b} \to 2\_aaaa \\
3 & 1\underline{a}aab \to 2bbb\underline{b} \to 2\_aaaa \\
4 & \left.\begin{matrix}1\\2\end{matrix}\right\} abb\underline{b} \to 2\underline{a}aaa \to 2bbbb\_ \\
\hline
 & \texttt{length} = 5 \\
1 & 3abbb\underline{b} \to 2\underline{a}aaab \to 2\_aaaaa \\
2 & 1\underline{a}aaab \to 1bbbb\underline{b} \to 2\_aaaaa \\
3 & 2\underline{a}aaab \to 2bbbb\underline{b} \to 2\_aaaaa \\
4 & \left.\begin{matrix}1\\2\end{matrix}\right\} abbb\underline{b} \to 2\underline{a}aaaa \to 1bbbbb\_ \\
5 & 3\underline{a}aabb \to 1abbb\underline{b} \to 1bbbbb\_ \\
\hline
 & \texttt{length} = 6 \\
1 & 3abbbb\underline{b} \to 2\underline{a}aaaab \to 2\_aaaaaa \\
2 & 2\underline{a}aaaab \to 1bbbbb\underline{b} \to 2\_aaaaaa \\
3 & 1\underline{a}aaaab \to 2bbbbb\underline{b} \to 2\_aaaaaa \\
4 & \left.\begin{matrix}1\\2\end{matrix}\right\} abbbb\underline{b} \to 2\underline{a}aaaaa \to 2bbbbbb\_ \\
5 & 3\underline{a}aabab \to 2abbbb\underline{b} \to 2bbbbbb\_ \\
\hline
 & \texttt{length} = 7 \\
1 & 3ab^5\underline{b} \to 2\underline{a}a^5b \to 2\_a^7 \\
2 & 1\underline{a}a^5b \to 1b^6\underline{b} \to 2\_a^7 \\
3 & 2\underline{a}a^5b \to 2b^6\underline{b} \to 2\_a^7 \\
4 & \left.\begin{matrix}1\\2\end{matrix}\right\} ab^5\underline{b} \to 2\underline{a}a^6 \to 1b^7\_ \\
5 & 3\left.\begin{matrix}\underline{a}aabaab\\\underline{a}aaabbb\end{matrix}\right\} \to 1ab^5\underline{b} \to 1b^7\_ \\
\hline
 & \texttt{length} = 8 \\
1 & 3ab^6\underline{b} \to 2\underline{a}a^6b \to 2\_a^8 \\
2 & 2\underline{a}a^6b \to 1b^7\underline{b} \to 2\_a^8 \\
3 & 1\underline{a}a^6b \to 2b^7\underline{b} \to 2\_a^8 \\
4 & \left.\begin{matrix}1\\2\end{matrix}\right\} ab^6\underline{b} \to 2\underline{a}a^7 \to 2b^8\_ \\
5 & 3\left\{\begin{matrix}\underline{a}aabaaab\\\underline{a}aaabbab\\\underline{a}aaababb\end{matrix}\right\} \to 2ab^6\underline{b} \to 2b^8\_
\end{array}
\tag{124}
$$

and the number of distinct extra IGR's needed to obtain the IRR's of length 2, 3, etc. are : 5 3 1 2 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16. The strongly connected components with more than one element are just two with

$$\begin{aligned}\mathtt{SCC1} &= \begin{cases} \mathtt{1}\underline{\mathtt{T_1}} \to\to \mathtt{2\_T_2} \overset{\mathtt{b}}{\Rightarrow} \mathtt{2}\underline{\mathtt{a}}\mathtt{T_1} \to\to \mathtt{2\_aT_2} \\ \mathtt{2}\underline{\mathtt{T_1}} \to\to \mathtt{2\_T_2} \overset{\mathtt{b}}{\Rightarrow} \mathtt{1}\underline{\mathtt{a}}\mathtt{T_1} \to\to \mathtt{2\_aT_2} \end{cases} . \\ \mathtt{SCC2} &= \begin{cases} \mathtt{2}\underline{\mathtt{T_1}} \to\to \mathtt{1T_{2\_}} \overset{\mathtt{a}}{\Rightarrow} \mathtt{2T_1}\underline{\mathtt{b}} \to\to \mathtt{2T_2b\_} \\ \mathtt{2}\underline{\mathtt{T_1}} \to\to \mathtt{2T_{2\_}} \overset{\mathtt{a}}{\Rightarrow} \mathtt{2T_1}\underline{\mathtt{b}} \to\to \mathtt{1T_2b\_} \end{cases} \end{aligned} \tag{125}$$

Before going any further it will be very convenient to introduce a symbol $p(\mathtt{1},\mathtt{2},n) = \left\{\begin{matrix}\mathtt{1}\ n \text{ odd} \\ \mathtt{2}\ n \text{ even}\end{matrix}\right\}$ where $p$ stands for parity and the arguments are the *symbols* for the TM state not the integers 1 and 2 and so this will be used in place of a particular TM state. From the computer program output, the IGR's appear to be infinite in number as well as the IRR's but with a more complicated structure. Thus it seems to indicate that the IRR's are simpler to characterise than the IGR's, so the analysis will start there. From the computer output, there seems to be a general result (**??**) for the IRR($n$) which is as follows:

$$\begin{array}{ll} \left.\begin{matrix}\mathtt{1}\\ \mathtt{2}\end{matrix}\right\} \mathtt{ab}^{n-2}\underline{\mathtt{b}} \to \mathtt{2}\underline{\mathtt{a}}\mathtt{a}^{n-1} \to p(\mathtt{1},\mathtt{2},n)\mathtt{b}^n\mathtt{\_} & n \geq \mathtt{2} \\ \mathtt{3ab}^{n-2}\underline{\mathtt{b}} \to \mathtt{2}\underline{\mathtt{a}}\mathtt{a}^{n-2}\mathtt{b} \to \mathtt{2\_a}^n & n \geq \mathtt{3} \\ \mathtt{1}\underline{\mathtt{a}}\mathtt{a}^{n-2}\mathtt{b} \to p(\mathtt{1},\mathtt{2},n)\mathtt{b}^{n-1}\underline{\mathtt{b}} \to \mathtt{2\_a}^n & n \geq \mathtt{2} \\ \mathtt{2}\underline{\mathtt{a}}\mathtt{a}^{n-2}\mathtt{b} \to p(\mathtt{2},\mathtt{1},n)\mathtt{b}^{n-1}\underline{\mathtt{b}} \to \mathtt{2\_a}^n & n \geq \mathtt{2} \\ \mathtt{3}\underline{\mathtt{a}}s_{n-1}(\mathtt{a},\mathtt{b}) \to p(\mathtt{1},\mathtt{2},n)\mathtt{ab}^{n-2}\underline{\mathtt{b}} \to p(\mathtt{1},\mathtt{2},n)\mathtt{b}^n\mathtt{\_} & n \geq \mathtt{5} \end{array} \tag{126}$$

where $s_{n-1}(\mathtt{a},\mathtt{b})$ is the set of all sequences of $\mathtt{a}$ and $\mathtt{b}$ of length $n-\mathtt{1}$ such that $\mathtt{3}\underline{\mathtt{a}}s_{n-1}(\mathtt{a},\mathtt{b}) \to p(\mathtt{1},\mathtt{2},n)\mathtt{ab}^{n-2}\underline{\mathtt{b}}$ but is not characterised any further yet.

*Proof.* To prove this by induction using results for $n \leq \mathtt{5}$ that are easily checked in the computer output Table **??**: first `F` needs to applied to (**??**).1,3,4 only, because (**??**).2 and (**??**).5 are not of extendable type. After proving these by induction, then the others will follow. The backward search gives, writing each step unless otherwise indicated:

$$\mathtt{1ab}^{n-2}\underline{\mathtt{b}}\alpha \begin{cases} \overset{\alpha=\mathtt{b}}{\leftarrow} \mathtt{3ab}^{n-2}\mathtt{b}\underline{\mathtt{b}} \\ \leftarrow \mathtt{2ab}^{n-3}\underline{\mathtt{a}}\mathtt{b}\alpha \leftarrow \mathtt{1ab}^{n-4}\underline{\mathtt{a}}\mathtt{ab}\alpha \leftarrow \mathtt{2ab}^{n-5}\underline{\mathtt{a}}\mathtt{aab}\alpha \ldots \mathtt{p}(\mathtt{2},\mathtt{1},n)\mathtt{a}\underline{\mathtt{a}}\mathtt{a}^{n-3}\mathtt{b}\alpha \leftarrow \emptyset \end{cases} . \tag{127}$$

Likewise

$$2\mathtt{ab}^{n-2}\underline{\mathtt{b}}\alpha\begin{cases}\overset{\alpha=\mathtt{a}}{\leftarrow}\begin{Bmatrix}1\\2\end{Bmatrix}\mathtt{ab}^{n-2}\mathtt{b}\underline{\mathtt{b}}\\ \leftarrow 1\mathtt{ab}^{n-3}\underline{\mathtt{a}}\mathtt{b}\alpha\leftarrow\begin{cases}2\mathtt{ab}^{n-4}\underline{\mathtt{a}}\mathtt{ab}\alpha\leftarrow\begin{cases}1\mathtt{ab}^{n-5}\underline{\mathtt{a}}\mathtt{aab}\alpha(1)\\ \begin{Bmatrix}1\\2\end{Bmatrix}\mathtt{ab}^{n-4}\mathtt{a}\underline{\mathtt{b}}\mathtt{b}\alpha\begin{cases}\overset{1}{\leftarrow}3\mathtt{ab}^{n-4}\mathtt{ab}\underline{\mathtt{b}}\alpha\leftarrow\emptyset\\ \overset{2}{\leftarrow}\emptyset\end{cases}\end{cases}\\ 3\mathtt{ab}^{n-3}\mathtt{a}\underline{\mathtt{b}}\alpha\leftarrow 3\mathtt{ab}^{n-3}\underline{\mathtt{a}}\mathtt{b}\alpha\leftarrow\emptyset\end{cases}\end{cases}\tag{128}$$

Here there is a cycle of length 2 that ends at (1) when $n$ is odd by reaching $1\mathtt{a}\underline{\mathtt{a}}\mathtt{a}^{n-3}\mathtt{b}\alpha$ from which no further backward steps are possible. If $n$ is even it is very complicated. The results (**??**) and (**??**) can be summarised by

$$\begin{aligned}&1\mathtt{ab}^{n-2}\underline{\mathtt{b}}\alpha\overset{\mathtt{b}}{\leftarrow}3\mathtt{ab}^{n-1}\underline{\mathtt{b}}\\ &2\mathtt{ab}^{n-2}\underline{\mathtt{b}}\alpha\begin{cases}\leftarrow\begin{Bmatrix}1\\2\end{Bmatrix}\mathtt{ab}^{n-1}\underline{\mathtt{b}}\\ \overset{n\ \text{even}}{\leftarrow}3\underline{\mathtt{a}}s_{n-1}(\mathtt{a},\mathtt{b})\alpha\\ \overset{n\ \text{odd}}{\leftarrow}\emptyset\end{cases}\end{aligned}\tag{129}$$

The first of these gives the IRR (**??**).2 with $n$ increased by 1 and the second gives both parts of (**??**).1 with $n$ increased by 1 using the following results (**??**) (that are very easy to show) to get the RHS's:

$$\begin{aligned}&1\mathtt{b}^n\underline{\mathtt{a}}\rightarrow 2\mathtt{b}^{n+1}\_\\ &1\mathtt{b}^n\underline{\mathtt{b}}\rightarrow 2\_\mathtt{a}^{n+1}\\ &2\mathtt{b}^n\underline{\mathtt{a}}\rightarrow 1\mathtt{b}^{n+1}\_\\ &2\mathtt{b}^n\underline{\mathtt{b}}\rightarrow 2\_\mathtt{a}^{n+1}\\ &2\underline{\mathtt{b}}\mathtt{a}^n\rightarrow 2\_\mathtt{a}^{n+1}\\ &2\underline{\mathtt{a}}\mathtt{a}^n\rightarrow p(2,1,n)\mathtt{b}^{n+1}\_\end{aligned}\tag{130}$$

together with $p(2,1,n)=p(1,2,n+1)$. Applying $\mathtt{F}$ to (**??**).3 gives (**??**).4 with $n\rightarrow n+1$ and vice versa with $\alpha=\mathtt{b}$. Putting $\alpha=\mathtt{a}$ on the left of (**??**).4 gives $p(1,2,n+1)\mathtt{ab}^{n-1}\underline{\mathtt{b}}\rightarrow 2\underline{\mathtt{a}}\mathtt{a}^n\rightarrow p(1,2,n+1)\mathtt{b}^{n+1}\_$ which contains the right hand half of (**??**).5 and because the conditions of Lemma **??** are satisfied it is an IRR of type RLR with $n\rightarrow n+1$. This together with (**??**) establishes (**??**) by induction. □

These middle elements (LHS) in(**??**) were included for clarity. They are a record of the successive values of $\alpha$ used in the chain of applications of $\mathtt{F}$ starting from the single TM steps. In this proof, the middle elements just need $\alpha$ to be added at the appropriate end, and the new RHS is just found from the original RHS and $\alpha$ by taking the computation as far as possible.

These results for the IRR's can now be expressed in terms of the IGR's that generate them. These IGR's were implicit in the above proof.

Directly from (**??**).2 with $\alpha$ added gives

$$2\alpha\underline{\mathtt{a}}\mathtt{a}^{n-2}\begin{cases} \leftarrow 3\alpha\mathtt{a}\mathtt{b}^{n-2}\underline{\mathtt{b}} \\ \overset{\mathtt{b}}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{a}^{n-1}\mathtt{b} \end{cases} \tag{131}$$

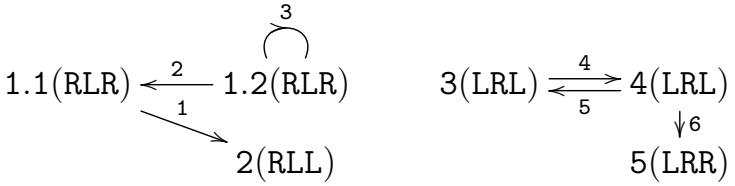


Figure 8: **The relationship between the IGR's in** (**??**) **and the parts of** (**??**) **for consecutive values of** $n$. The nodes in larger font are IRR's in (**??**) numbered consecutively and the type of the IRR appears in parentheses after the IRR number. Each arrow corresponds to an IGR in (**??**) including the one relating (**??**).1.2 to itself and are also numbered consecutively. IGR's 1 and 6 generate IRR's of non-extendable type. The two disconnected parts correspond to `SCC2` and `SCC1` respectively. To these the two relationships in (**??**).{7,8} must be added.

The IGR's are as follows in (**??**). The last two don't involve $\mathtt{T}_1$ and $\mathtt{T}_2$ because each is used only in a single context as shown given by specifying and including the values of the arbitrary strings into the IGR's. If there are more than one array in an IGR, the top values go together to get one result, and the bottom ones to get another result.

$$
\begin{array}{l}
1\underline{\mathtt{T}_1} \rightarrow\!\rightarrow \left\{\begin{matrix}1\\2\end{matrix}\right\} \mathtt{T}_2\mathtt{b}^n\_ \stackrel{\mathtt{b}}{\Rightarrow} 3\mathtt{T}_1\underline{\mathtt{b}} \rightarrow\!\rightarrow 2\underline{\mathtt{T}_2}\mathtt{a}^{n+1} \text{ for } n \geq 0\\
2\underline{\mathtt{T}_1} \rightarrow\!\rightarrow \left\{\begin{matrix}1\\2\end{matrix}\right\} \mathtt{T}_2\_ \stackrel{\mathtt{a}}{\Rightarrow} 1\mathtt{T}_1\underline{\mathtt{b}} \rightarrow\!\rightarrow \left\{\begin{matrix}2\\1\end{matrix}\right\} \mathtt{T}_2\mathtt{b}\_\\
2\underline{\mathtt{T}_1} \rightarrow\!\rightarrow \left\{\begin{matrix}1\\2\end{matrix}\right\} \mathtt{T}_2\_ \stackrel{\mathtt{a}}{\Rightarrow} 2\mathtt{T}_1\underline{\mathtt{b}} \rightarrow\!\rightarrow \left\{\begin{matrix}2\\1\end{matrix}\right\} \mathtt{T}_2\mathtt{b}\_\\
1\underline{\mathtt{T}_1} \rightarrow\!\rightarrow 2\_\mathtt{T}_2 \stackrel{\mathtt{b}}{\Rightarrow} 2\underline{\mathtt{a}}\mathtt{T}_1 \rightarrow\!\rightarrow 2\_\mathtt{a}\mathtt{T}_2\\
2\underline{\mathtt{T}_1} \rightarrow\!\rightarrow 2\_\mathtt{T}_2 \stackrel{\mathtt{b}}{\Rightarrow} 1\underline{\mathtt{a}}\mathtt{T}_1 \rightarrow\!\rightarrow 2\_\mathtt{a}\mathtt{T}_2\\
2\underline{\mathtt{a}}\mathtt{a}^{n-2}\mathtt{b}\mathtt{T}_1 \rightarrow\!\rightarrow 2\_\mathtt{a}^n\mathtt{T}_2 \stackrel{\mathtt{a}}{\Rightarrow}\\
3\underline{\mathtt{a}}s_{n-1}(\mathtt{a},\mathtt{b})\mathtt{T}_1 \rightarrow\!\rightarrow \mathtt{p}(1,2,n+1)\mathtt{b}^{n+1}\underline{\mathtt{T}_2} \text{ for } n \geq 4\\
3\underline{\mathtt{b}} \rightarrow\!\rightarrow 1\_\mathtt{b} \stackrel{\mathtt{a}}{\Rightarrow} 3\underline{\mathtt{a}}\mathtt{b} \rightarrow\!\rightarrow 2\_\mathtt{a}\mathtt{a}\\
3\underline{\mathtt{a}}\mathtt{b} \rightarrow\!\rightarrow 2\_\mathtt{a}\mathtt{a} \stackrel{\mathtt{a}}{\Rightarrow} 3\underline{\mathtt{a}}\mathtt{a}\mathtt{b} \rightarrow\!\rightarrow 1\mathtt{b}\mathtt{b}\mathtt{b}\_
\end{array}
\quad . \tag{132}
$$

Therefore the LIGR's found are as follows:

$$
\begin{array}{l}
1\underline{\mathtt{T}} \stackrel{\mathtt{b}}{\leftarrow} 3\mathtt{T}\underline{\mathtt{b}}\\
2\underline{\mathtt{T}} \stackrel{\mathtt{a}}{\leftarrow} 1\mathtt{T}\underline{\mathtt{b}}\\
2\underline{\mathtt{T}} \stackrel{\mathtt{a}}{\leftarrow} 2\mathtt{T}\underline{\mathtt{b}}\\
1\underline{\mathtt{T}} \stackrel{\mathtt{b}}{\leftarrow} 2\underline{\mathtt{a}}\mathtt{T}\\
2\underline{\mathtt{T}} \stackrel{\mathtt{b}}{\leftarrow} 1\underline{\mathtt{a}}\mathtt{T}\\
2\underline{\mathtt{a}}\mathtt{a}^{n-2}\mathtt{b}\mathtt{T} \stackrel{\mathtt{a}}{\leftarrow} 3\underline{\mathtt{a}}s_{n-1}(\mathtt{a},\mathtt{b})\mathtt{T}\\
3\underline{\mathtt{b}} \stackrel{\mathtt{a}}{\leftarrow} 3\underline{\mathtt{a}}\mathtt{b}\\
3\underline{\mathtt{a}}\mathtt{b} \stackrel{\mathtt{a}}{\leftarrow} 3\underline{\mathtt{a}}\mathtt{a}\mathtt{b}
\end{array}
\quad . \tag{133}
$$

The result (**??**).6 can, from the computer results for $n = 4$ and 5, be shortened by absorbing the $\mathtt{b}$ on the extreme left into the arbitrary string $\mathtt{T}_1$ because the pointer does not reach that far during the computation. Attempting to prove this generally seems a bit difficult.

The LIGR's found by the computer program when $n$ is 6 are as follows:

$$
\begin{array}{l}
1\underline{\mathtt{T}} \leftarrow 3\mathtt{T}\underline{\mathtt{b}}\\
2\underline{\mathtt{T}} \leftarrow \left\{\begin{matrix}1\mathtt{T}\underline{\mathtt{b}}\\2\mathtt{T}\underline{\mathtt{b}}\end{matrix}\right.\\
1\underline{\mathtt{T}} \leftarrow 2\underline{\mathtt{a}}\mathtt{T}\\
2\underline{\mathtt{T}} \leftarrow 1\underline{\mathtt{a}}\mathtt{T}\\
2\underline{\mathtt{a}}\mathtt{a}\mathtt{a}\mathtt{T} \leftarrow 3\underline{\mathtt{a}}\mathtt{a}\mathtt{a}\mathtt{b}\mathtt{T}\\
3\underline{\mathtt{T}} \leftarrow 3\underline{\mathtt{a}}\mathtt{T}
\end{array}
\tag{134}
$$

However by combining the results of **??** with the single steps of the TM itself, it is quite easy to come up with the following description of the TM:

Table 12: Equivalent description of TM **??**

| A: $\mathtt{2\_a}^n$ | B: $\mathtt{1b}^n\mathtt{\_}$ | C: $\mathtt{2b}^n\mathtt{\_}$ | D: $\mathtt{3a}^n\mathtt{\_}$ |
|---|---|---|---|
| $\mathtt{a} \to$ B†($n$ even) | $\mathtt{a} \to$ C | $\mathtt{a} \to$ B | $\mathtt{a} \to$ D |
| $\mathtt{a} \to$ C†($n$ odd) | $\mathtt{b} \to$A† | $\mathtt{b} \to$ A† | $\mathtt{b} \to$ B*. |
| $\mathtt{b} \to$ A | | | |

Here each column corresponds to the CS at its head labelled from A to D. The meaning of the entries in the body of the table is that if the symbol on the left is at the pointer in the CS (extending the string by one symbol), the result of the following computation is the CS indicated on the right with $n$ replaced by $n$+1. The symbol †indicates that the pointer swaps sides and the result at * is actually $\mathtt{1a}^{n-2}\mathtt{bbb\_}$ if $n > 2$ so that the new $n$ is now 3. The other cases are dealt with by $\mathtt{3\underline{a}b} \to \mathtt{3a\underline{b}} \to \mathtt{2\_aa}$, $\mathtt{3b\underline{b}} \to \mathtt{2\_ab}$, and $\mathtt{3\underline{a}aab} \to \mathtt{3aa\underline{b}} \to \mathtt{1bbb\_}$. Notice that when the pointer swaps ends going right, only b's are produced, and if it swaps ends and goes left only a's are produced. Also the TM cannot reach a configuration in D by steps in Table **??** unless it started from a configuration in D, so its description should start from D. As long as the new symbol is a the TM continues to the right. If a b is reached, it goes to a configuration in B, from where successive a's alternate between CS's B and C. If a b is reached in either B or C then it swaps sides leaving a's and getting to A. Then more b's just leave it in A, but an a makes it swap sides to B or C according to whether $n$ is even or odd respectively. Thus the description of TM (**??**) is clearly explained as an expanding cycle, which to me seems a satisfactory place to leave the general analysis of this example.

In general the question remains that because IRR (**??**).2 cannot generate any new IRR's with F, how does this result relate to others to which it must relate because of neighbouring symbols. For example if a is added on the left and the computation is continued with **??** it gives
$\mathtt{3aab}^{n-2}\mathtt{\underline{b}} \to \mathtt{2a\underline{a}a}^{n-2}\mathtt{b} \to \mathtt{2\underline{a}a}^{n} \to p(2,1,n)\mathtt{b}^{n+1}\mathtt{\_}$ which (ignoring the intermediate step) by lemma **??** is not an IRR (at least for $n = 3$). Putting $\mathtt{b}^k$ on the left of (**??**).2 gives $\mathtt{3b}^k\mathtt{ab}^{n-1}\mathtt{\underline{b}} \to\!\!\to \mathtt{2b}^{k-1}\mathtt{\underline{b}a}^{n+1} \to \mathtt{2\_a}^{n+k+1}$ and again putting a on the left gives $\mathtt{3ab}^k\mathtt{ab}^{n-1}\mathtt{\underline{b}} \to \mathtt{2\underline{a}a}^{n+k+1} \to p(2,1,n+k+1)\mathtt{b}^{n+k+2}\mathtt{\_}$. This shows that there can be other methods than F for generating IRR's from others that increase the length of the IRR by 1.

Following up on this idea gives an apparently general method of analysis that seems to generalise (**??**). Let X be an IRR of length $n$ of type RLR then it is extendable by F as in Theorem **??** provided a suitable added symbol $\alpha$ can be found giving an IRR of type RLL or RLR. Alternatively if X is of type RLL i.e. of type $\mathtt{n} \to \mathtt{1} \to \mathtt{0}$ (see the notation of Section **??**) then the computation can be continued after adding a symbol $\alpha$ on the left to get either one of the forms $\mathtt{n} \to \mathtt{1} \to \mathtt{0} \to \mathtt{n}+\mathtt{1}$ or $\mathtt{n} \to \mathtt{1} \to \mathtt{0} \to -\mathtt{1}$. In either case there is a CS $\mathtt{n}'$ (which could be the same as n) and is the last to be arrived at out of the CS's (if there are more than one of them) that occur with the

pointer at position $n$ before the CS 0 is reached. Therefore the computation can be written as $\mathtt{n} \to \mathtt{n}'$ followed by either $\mathtt{Y1} = \mathtt{n}' \to 0 \to \mathtt{n}+1$ (type RLR) or $\mathtt{Y2} = \mathtt{n}' \to 0 \to -1$ (type RLL). The only way of avoiding either of these possibilities (in both cases when X has type RLR or type RLL) is if the computation after the CS 0 continues indefinitely with the pointer in the range $[0, n]$. In this case because there are a finite number of CS's involved, the sequence of CS's representing the computation must eventually repeat a CS therefore a stationary cycle occurs after the computation reaches CS 0. There is no other CS 0 entered before the CS's 0 given here because these are derived from X where it is reached by a single TM step, therefore by Lemma **??**, Y1 and Y2 are also IRR's of length $n+1$. Here the CS $\mathtt{n}'$ is uniquely determined by n so Y1 or Y2 is uniquely determined once X is fixed, and this works for every value of $\alpha$.

This result in either of these other cases could be called F(X) where F has now been extended to apply to IRR's that were formerly non-extendable i.e. those of types RLL and LRR. The result F(X) is another IRR of length $n+1$ which, in the case that X is of the type RLL or LRR, is preceded by the computation $\mathtt{n} \to \mathtt{n}'$ starting and ending at the same point. The IRR's obtained like this from IRR's of type RLL or LRR cannot generate any new IRR's that are not already obtainable from IRR's of type LRL or RLR because these are already known to be the whole set of IRR's by Theorem **??**.

The result of the previous paragraph (and of course its right-left reversed form) can be combined with the results of Section **??** to show how an infinite sequence of IRR's of lengths increasing by one at each step can be derived that is prevented only the presence of stationary cycles or the absence of new origins. They can be chained together because the truncations can be undone. In this sequence the IRR's that start on the right, symbols are added on the left by G and on the right by F and if they started on the left it would of course be reversed. Therefore the pointer generally goes back and forth with ever increasing sweeps across the string of symbols on the tape (though it may not happen in every case), extending in both directions if both F and G continue to be involved.

These results for IRR's starting on the right can be symbolised as follows:

$$\begin{aligned}&\exists\mathtt{RLR}(n) \Rightarrow \forall(\alpha \text{ used by the TM})\{\text{no new origin found using } \alpha \text{ in } \mathtt{F} \vee \\ &\qquad \exists\mathtt{RLR}(n+1) \vee \exists\mathtt{RLL}(n+1) \vee \exists \text{ stationary cycle}\} \\ &\exists\mathtt{RLL}(n) \Rightarrow \exists\mathtt{RLR}(n+1) \vee \exists\mathtt{RLL}(n+1) \vee \exists \text{ a stationary cycle}\end{aligned} \tag{135}$$

An equivalent (mirror image) idea relating IRR's of types LRL, LRR exists which should generate the same thing from the other end ie. IRR's starting at the left. These are both approaching the description of the TM behaviour as a sequence of expanding cycles from opposite ends but represent the same reality. Involved in general (as shown by the analysis of TM **??**) are both stationary cycles where the TM effectively stops, and sequences of symbols

which the reversed TM cannot cross which can prevent a new origin being found. It seems to me that this will generate the interpretive cycle of a TM that has been shown to have this behaviour.

# References

[1] Methods for Understanding Turing Machine Computations

[2] Reverse engineering Turing Machines and the Collatz Conjecture

[3] The previous version in D of the computer program for analysis of Turing Machines

[4] Program for just doing the backward search for a single CS

[5] The new program tie v3.2 for doing the computations in this paper

Anything beyond this point is probably not needed but is kept just in case. It will be removed to another document later.

**Lemma 8.1.** *The backward search from any CS of the form* $\mathtt{1Tadb\underline{a}aa}\alpha$ *cannot lead to any new LIGR's or RCS's provided the string* `T` *contains the symbol* `a`. ********* not needed *************

*Proof.* This is by continuing the backward search from there. This gives the following tree

$$\mathtt{1Tadb\underline{a}aa}\alpha \leftarrow \begin{cases} \mathtt{1Tad\underline{c}a^3}\alpha \leftarrow \begin{cases} \mathtt{3Tadc\underline{d}a^2}\alpha \leftarrow \mathtt{3Tad\underline{c}da^2}\alpha \leftarrow \begin{cases} \mathtt{2Ta\underline{a}cda^2}\alpha \\ \mathtt{1Tadc\underline{b}a^2}\alpha * \end{cases} \\ \mathtt{1Ta\underline{c}ca^3}\alpha \leftarrow \mathtt{2Tac\underline{c}a^3}\alpha \leftarrow \mathtt{2Ta\underline{b}ca^3}\alpha \end{cases} \\ \mathtt{3Tadba\underline{d}a}\alpha \end{cases} \tag{136}$$

$$\mathtt{1Tadc\underline{b}a^2}\alpha \leftarrow \mathtt{3Tadcb\underline{d}a}\alpha \leftarrow \mathtt{2Tadc\underline{a}da}\alpha \leftarrow \mathtt{2Tad\underline{b}ada}\alpha \leftarrow \mathtt{1Ta\underline{a}bada}\alpha \tag{137}$$

where the computation stopped whenever either no reverse TM step is possible, or when by Lemmas (**??**) or (**??**) the pointer cannot go beyond the string as a result of continued backward searching. Because all branches of the tree do eventually lead to a halt, no LIGR's or RCS's can result from further backward searching. □

**Lemma 8.2.** ************* not needed ************ Backward searching starting from any CS of the form* $\mathtt{1Td\underline{c}abada}\alpha$ *leads to exactly the following*

*set of CS's regardless of the arbitrary string* `T` *in addition to possible CS's with the pointer at the left depending on* `T`*:*

$$\begin{array}{l}
\texttt{1Tdca}^{2}\texttt{dbd}\underline{\texttt{b}} \\
\texttt{3Tdca}^{3}\texttt{db}\underline{\texttt{d}} \\
\texttt{2Tdca}^{3}\texttt{db}\underline{\texttt{c}} \\
\texttt{1Tdcdbdbd}\underline{\texttt{b}} \\
\texttt{3Tdcdcbdb}\underline{\texttt{d}} \\
\texttt{2Tdcdcbdb}\underline{\texttt{c}} \\
\texttt{3Tdcdbadb}\underline{\texttt{d}} \\
\texttt{2Tdcdbadb}\underline{\texttt{c}}
\end{array} \tag{138}$$

*These are related to the set of LIGR's in* (**??**).15-19.

*Proof.* The backward search stops if either (1) the pointer can be shown not to get to the right because of `cx` on the right of the pointer or (2) no further backward TM steps are possible or (3) the end of the known symbols on the string is reached or (4) a stationary cycle is reached. The numbers after * indicate continuations.

$$1\mathtt{Td}\underline{\mathtt{c}}\mathtt{abada}\alpha \leftarrow \begin{cases} 1\mathtt{T}\underline{\mathtt{c}}\mathtt{cabada}\alpha \\ 3\mathtt{Tdc}\underline{\mathtt{d}}\mathtt{bada}\alpha \leftarrow \begin{cases} 3\mathtt{Td}\underline{\mathtt{c}}\mathtt{dbada}\alpha \leftarrow \begin{cases} 2\mathtt{T}\underline{\mathtt{a}}\mathtt{cdbada}\alpha \\ 1\mathtt{Tdc}\underline{\mathtt{b}}\mathtt{bada}\alpha \end{cases} \\ 1\mathtt{Tdcd}\underline{\mathtt{b}}\mathtt{ada}\alpha \leftarrow \begin{cases} 1\mathtt{Tdc}\underline{\mathtt{c}}\mathtt{bada}\alpha \\ 3\mathtt{Tdcdb}\underline{\mathtt{d}}\mathtt{da}\alpha * 1 \end{cases} \end{cases} \end{cases}$$

$$*1 \leftarrow \begin{cases} 2\mathtt{Tdcd}\underline{\mathtt{a}}\mathtt{dda}\alpha \leftarrow 1\mathtt{Tdc}\underline{\mathtt{a}}\mathtt{adda}\alpha \leftarrow 3\mathtt{Tdca}\underline{\mathtt{d}}\mathtt{dda}\alpha \leftarrow 1\mathtt{Tdcad}\underline{\mathtt{b}}\mathtt{da}\alpha * 2 \\ 1\mathtt{Tdcdbd}\underline{\mathtt{b}}\mathtt{a}\alpha * 5 \end{cases}$$

$$*2 \leftarrow \begin{cases} 1\mathtt{Tdca}\underline{\mathtt{c}}\mathtt{bda}\alpha \\ 3\mathtt{Tdcadb}\underline{\mathtt{d}}\mathtt{a}\alpha \leftarrow 2\mathtt{Tdcad}\underline{\mathtt{a}}\mathtt{da}\alpha \leftarrow 1\mathtt{Tdca}\underline{\mathtt{a}}\mathtt{ada}\alpha \leftarrow 3\mathtt{Tdcaa}\underline{\mathtt{d}}\mathtt{da}\alpha * 3 \end{cases}$$

$$*3 \leftarrow 1\mathtt{Tdcaad}\underline{\mathtt{b}}\mathtt{a}\alpha \leftarrow \begin{cases} 1\mathtt{Tdcaa}\underline{\mathtt{c}}\mathtt{ba}\alpha \\ 3\mathtt{Tdcaadb}\underline{\mathtt{d}}\alpha \leftarrow \begin{cases} 2\mathtt{dca}^2\mathtt{d}\underline{\mathtt{a}}\mathtt{d}\alpha \leftarrow 1\mathtt{Tdcaa}\underline{\mathtt{a}}\mathtt{ad}\alpha * 4 \\ 1\mathtt{Tdca}^2\mathtt{dbd}\underline{\mathtt{b}} \end{cases} \end{cases}$$

$$*4 \leftarrow 3\mathtt{Tdca}^3\underline{\mathtt{d}}\mathtt{d}\alpha \leftarrow 1\mathtt{Tdca}^3\mathtt{d}\underline{\mathtt{b}}\alpha \leftarrow \begin{cases} 1\mathtt{Tdca}^3\underline{\mathtt{c}}\mathtt{b}\alpha \\ 3\mathtt{Tdca}^3\mathtt{db}\underline{\mathtt{d}} \\ 2\mathtt{Tdca}^3\mathtt{db}\underline{\mathtt{c}} \end{cases}$$

$$*5 \leftarrow \begin{cases} 1\mathtt{Tdcdb}\underline{\mathtt{c}}\mathtt{ba}\alpha \leftarrow 1\mathtt{Tdcd}\underline{\mathtt{c}}\mathtt{cba}\alpha \\ 3\mathtt{Tdcdbdb}\underline{\mathtt{d}}\alpha \leftarrow \begin{cases} 2\mathtt{Tdcdbd}\underline{\mathtt{a}}\mathtt{d}\alpha \leftarrow 1\mathtt{Tdcdb}\underline{\mathtt{a}}\mathtt{ad}\alpha \leftarrow \begin{cases} 1\mathtt{Tdcd}\underline{\mathtt{c}}\mathtt{aad}\alpha * 6 \\ 3\mathtt{Tdcdba}\underline{\mathtt{d}}\mathtt{d}\alpha * 8 \end{cases} \\ 1\mathtt{Tdcdbdbd}\underline{\mathtt{b}} \end{cases} \end{cases}$$

$$*6 \leftarrow \begin{cases} 1\mathtt{Tdc}\underline{\mathtt{c}}\mathtt{caad}\alpha \\ 3\mathtt{Tdcdc}\underline{\mathtt{d}}\mathtt{ad}\alpha \leftarrow 3\mathtt{Tdcd}\underline{\mathtt{c}}\mathtt{dad}\alpha \leftarrow \begin{cases} 2\mathtt{Tdc}\underline{\mathtt{a}}\mathtt{cdad}\alpha \\ 1\mathtt{Tdcdc}\underline{\mathtt{b}}\mathtt{ad}\alpha \leftarrow 3\mathtt{Tdcdcb}\underline{\mathtt{d}}\mathtt{d}\alpha * 7 \end{cases} \end{cases}$$

$$*7 \leftarrow \begin{cases} 2\mathtt{Tdcdc}\underline{\mathtt{a}}\mathtt{dd}\alpha \leftarrow 2\mathtt{Tdcd}\underline{\mathtt{b}}\mathtt{add}\alpha \leftarrow 1\mathtt{Tdc}\underline{\mathtt{a}}\mathtt{badd}\alpha \\ 1\mathtt{Tdcdcbd}\underline{\mathtt{b}}\alpha \leftarrow \begin{cases} 1\mathtt{Tdcdcb}\underline{\mathtt{c}}\mathtt{b}\alpha \leftarrow 1\mathtt{Tdcdc}\underline{\mathtt{c}}\mathtt{cb}\alpha \\ 3\mathtt{Tdcdcbdb}\underline{\mathtt{d}} \\ 2\mathtt{Tdcdcbdb}\underline{\mathtt{c}} \end{cases} \end{cases}$$

$$*8 \leftarrow 1\mathtt{Tdcdbad}\underline{\mathtt{b}}\alpha \leftarrow \begin{cases} 1\mathtt{Tdcdba}\underline{\mathtt{c}}\mathtt{b}\alpha \\ 3\mathtt{Tdcdbadb}\underline{\mathtt{d}} \\ 2\mathtt{Tdcdbadb}\underline{\mathtt{c}} \end{cases}$$

(139)

□

Note forward reference to Table **??**.

**Lemma 8.3.** ******* not needed ********* If in a row of Table* **??***, some*

*LIGR's are produced then in another row with the RHS of "its affect" differing only by the symbol next to the string* $\mathtt{T}$*, the same set of LIGR's is produced.*

*Proof.* Suppose a backward search gives

$$\mathtt{ST}\beta\underline{\mathtt{T_1}}\alpha \leftarrow \mathtt{A} \tag{140}$$

in the Table **??** where the pointer is at the right hand end of $\mathtt{T_1}$ where $\mathtt{A}$ is a set of RCS's and LIGR's. Then this will be based on

$$\mathtt{ST}\underline{\mathtt{T_1}}\alpha \leftarrow \mathtt{S_1T}\underline{\mathtt{T_2}}\alpha \tag{141}$$

(an RCS) that may be also in the same table where in (**??**) and (**??**) the pointer is at the left hand end of $\mathtt{T_2}$, and $\alpha$ and $\beta$ are arbitrary symbols (with $\alpha$ and $\mathtt{T}$ having their usual roles) and $\mathtt{T_1}$ and $\mathtt{T_2}$ (as is $\mathtt{T}$) are arbitrary strings of symbols and $\mathtt{S}$ and $\mathtt{S_1}$ are arbitrary states. This is because truncating the string to the right of $\mathtt{T}$ by one symbol on the left will convert any LIGR's arising only because of that last symbol to RCS's. This will be done several times if necessary to get the required line of the grand search. Therefore (**??**) can be written as

$$\mathtt{ST}\beta\underline{\mathtt{T_1}}\alpha \leftarrow \mathtt{S_1T}\beta\underline{\mathtt{T_2}}\alpha \leftarrow \mathtt{A}. \tag{142}$$

If the pointer does not reach $\beta$ in this derivation, it follows from this that $\beta$ can be replaced by any other symbol say $\gamma$

$$\mathtt{ST}\gamma\underline{\mathtt{T_1}}\alpha \leftarrow \mathtt{S_1T}\gamma\underline{\mathtt{T_2}}\alpha \leftarrow \{\mathtt{S_2T}\underline{\delta}\mathtt{T_2}\alpha, \mathtt{A}\}. \tag{143}$$

where the first member of the set on the right (an RCS) is there if and only if in the TM table $\mathtt{S_2}\underline{\delta} \rightarrow \mathtt{S_1}\gamma_{-}$, and $\gamma$ and $\delta$ are also arbitrary symbols. If (**??**) does not lead to any RCS's then the derivation cannot have the pointer reaching $\beta$ then the derivation is followed as in the proof of (**??**) except that $\beta$ is replaced by $\gamma$ and the pointer never reaches $\gamma$ leading to the same LIGR's after the unused symbol $\gamma$ has been removed and no RCS's. If $\beta$ is reached in (**??**) then follow the reverse steps that lead to the pointer reaching $\beta$ giving some RCS's that could be different from those in (**??**).

As long as $\beta$ is not reached by the pointer, the symbols to the right of $\beta$ are independent of $\beta$. Therefore if the backward search from (**??**) is completed, the corresponding results with a different value of $\beta$ are obtained by (1) assessing whether or not the single step to $\beta$ is possible from the start or from any point where the pointer reaches one space to the right of $\beta$ and if so including the RCS obtained, and (2) taking the results that don't take the pointer to $\beta$ and replacing $\beta$ by the new symbol. This leaves the LIGR's unchanged after the symbol in place of $\beta$ that plays no part in the calculations is removed. □

**Lemma 8.4.** *Doesn't seem to be needed but this looks usable.******** Any RCS of the form* $\mathtt{2T}\underline{\mathtt{b}}\mathtt{b}^{\mathtt{n}+1}\mathtt{add}\alpha$ *does not lead to any LIGR's for* $\mathtt{n} \geq \mathtt{0}$.

*Proof.* For convenience let the string $\mathtt{b}^{\mathtt{n}+1}\mathtt{add}\alpha$ be denoted by $\mathtt{S}$ because it remains the same throughout the proof and note that the leftmost symbol of $\mathtt{S}$ is $\mathtt{b}$ if $\mathtt{n} \geq 0$. Add an arbitrary symbol $\beta$ on the left according to the procedure described in section **??** and continue the backward search from there gives $2\mathtt{T}\beta\underline{\mathtt{b}}\mathtt{S} \leftarrow \begin{cases} 1\mathtt{T}\underline{\mathtt{a}}\mathtt{bS} \\ 2\mathtt{T}\underline{\mathtt{b}}\mathtt{bS} \end{cases}$. The second case is as above with $\mathtt{n}$ increased by 1. Repeat this for the first case giving $1\mathtt{T}\beta\underline{\mathtt{a}}\mathtt{bS} \leftarrow 1\mathtt{T}\underline{\mathtt{c}}\mathtt{abS}$. Repeat this argument again gives

$$1\mathtt{T}\beta\underline{\mathtt{c}}\mathtt{abS} \leftarrow \begin{cases} 1\mathtt{T}\underline{\mathtt{c}}\mathtt{cabS} * \\ 3\mathtt{T}\beta\mathtt{c}\underline{\mathtt{d}}\mathtt{bS} \leftarrow \begin{cases} 3\mathtt{T}\beta\underline{\mathtt{c}}\mathtt{dbS} \leftarrow \begin{cases} 1\mathtt{T}\beta\mathtt{c}\underline{\mathtt{b}}\mathtt{bS} \\ 2\mathtt{T}\underline{\mathtt{a}}\mathtt{cdbS} * \\ 3\mathtt{T}\underline{\mathtt{c}}\mathtt{cdbS} * \end{cases} \\ 1\mathtt{T}\beta\mathtt{cd}\underline{\mathtt{b}}\mathtt{S} \leftarrow 1\mathtt{T}\beta\mathtt{c}\underline{\mathtt{c}}\mathtt{bS} \end{cases} \end{cases} \tag{144}$$

In this search tree, $*$ indicates that by Lemma **??** the pointer can never reach $\alpha$ i.e. no new LIGR's can result from further additions of symbols. Because this search tree is complete it follows that the backward search from $2\mathtt{T}\underline{\mathtt{b}}\mathtt{b}^{\mathtt{n}+1}\mathtt{add}\alpha$ cannot lead to an LIGR unless the backward search from $2\mathtt{T}\underline{\mathtt{b}}\mathtt{b}^{\mathtt{n}+2}\mathtt{add}\alpha$ also leads to an LIGR. This gives an infinite regress showing that no RCS of this form can lead to an LIGR for $\mathtt{n} \geq 0$. □