Further Study of a Stack-length Model

D. T. GOODWIN*
University of Keele, Newcastle, Staffs., ST5 5BG

This paper continues analysis of the model of arithmetic stack behaviour set up in Goodwin (1977),[1] and further investigated in Goodwin (1980).[2] It studies a certain type of condition computable at compile time under which an infinity of program runs may use the stack without underflow.

1. INTRODUCTION

In Goodwin (1977) a model of the behaviour of arithmetic stack lengths at program execution time was proposed, and analysis was begun. This was continued in Goodwin (1980), and a further development is presented here. While the latter paper discussed conditions under which an infinity of different strings could all be accommodated on a stack of finite length, the present article begins to consider the same question when the stack may be indefinitely long, but while underflow is still not permitted.

The author is aware of two applications of the model under discussion, including arithmetic stack behaviour, and suspects and hopes that others may be found where the theory is of more practical use. Essentially it concerns the fluctuating numeric value of a single scalar quantity (cf. the number of cells on the stack), as it is affected by successive elements of a string which is somehow generated under the rules of a context-free grammar. For the arithmetic stack application it was shown that a program written in one of many programming languages could itself be regarded as a context-free grammar for the purpose of this work. Thus the examination of grammars discussed here corresponds to compile-time examination of the program text. Furthermore, a string generated by such a grammar corresponds to one particular run of a program, as it deposits and removes cells from the stack.

2. NOTATION AND CONCEPTS

This paper relies to a certain extent on the concepts and notation of the previous two, but a brief account of useful ideas is given here. Theorem and lemma numbers are continued. Theorems 1–6 and Lemmas 1–4 are in Goodwin (1977). Theorems 7–16 and Lemmas 5–6 are in Goodwin (1980).

In the examples of formal grammars and strings, non-terminals are denoted by capitals, and ordinary terminals and strings of symbols by small letters. Ordinary terminals are each deemed to deposit one item on the stack. The special terminal C removes one item from the stack. This could correspond to a transfer-to-store instruction in the run-time arithmetic stack application, or to an addition or other operation which combined two items on the stack into one. The particular grammar under discussion is called G.

A 'cycle' is used in the sense of a number of production-rule applications which together generate uSv from S, where u and v are strings of terminals and non-terminals. The 'length' of a cycle C is the number of cells that u and v together add to the stack. The left-hand length (lhl) of a cycle is the number of cells that u alone adds. These lengths may vary depending on the expansions into terminals of the non-terminals of u and v. $l^-(C)$ and $lhl^-(C)$ denote the lower bounds, if any, of all these lengths. 'Basic' cycles are a finite set of cycles from which all cycles may be generated.

* Address for correspondence: 2 West Lawn, Bearwood College, Wokingham, Berks., RG11 5BH.

3. SOME SUFFICIENT CONDITIONS FOR AN INFINITY OF STRINGS

Theorem 4 states that, if every basic cycle of a grammar G always has non-negative left-hand and total lengths, then there is a lower limit of stack length which is always exceeded by every string of G during its deposition on the stack. (The possibility of negative stack lengths is allowed – this could be practical if one string were being deposited on top of another on a physical stack.)

For example, for the grammar

$$S \rightarrow aSb$$
$$S \rightarrow f$$

the cycle C is $S \rightarrow aSb$, and the strings generated are $a^n f b^n$, for $n > 0$. These all have minimum length 1 during deposition. (This is attained after the first 'a' is deposited on the stack.) Here $lhl(C) = l(a) = 1$, and $l(C) = l(ab) = 2$. The whole infinity of strings generated by G are subject to the minimum length. However, the conditions of Theorem 4 can be considerably relaxed without removing the existence of an infinity of acceptable strings, although not all the strings of G would then have the property.

One relaxation of the conditions would be to stipulate just 'for at least one' of the basic cycles C of G that $lhl^-(C) >= 0$ and $l^-(C) >= 0$. Then the example grammar could be changed to

$$S \rightarrow aSb$$
$$S \rightarrow CS$$
$$S \rightarrow f$$

and there would still be the acceptable infinity of strings $a^n f b^n$, even though there would be others (e.g. the strings $C^n f$) with no lower stack bound.

A further relaxation would be that for the particular basic cycle C the given condition need not apply to all its terminal derivations. For example, S → MSN

M → a

M → **C**

N → b

N → **CC**

S → f.

This grammar has one cycle S → MSN.

lhl(C) = 1 if M → a

= −1 if M → **C**

l(C) = 2 if M → a **and** N → b,

but l(C) = −3 if M → **C** **and** N → **CC**.

Thus while lhl⁻(C) = −1 and l⁻(C) = −3 are both unacceptably low limits, nevertheless, if M → a and N → b, then the cycle still generates the acceptable infinity $a^n f b^n$, as before.

It is not even necessary for *any basic* cycle to have both its length and left-hand length non-negative; for example,

S → a**SCC**

S → **CS**bb

S → d.

Here each recursive rule defines a basic cycle, but neither satisfies both conditions. However, if applied alternately they form a non-basic or 'composed' cycle which generates the acceptable strings $(a\mathbf{C})^n d(bb\mathbf{CC})^n$. This latter kind of behaviour is the case where the stack is bounded above as well as below and Theorem 14 applies: 'G is infinitely stack-bounded in some interval $[L_1, L_2]$ if and only if there exists a cycle C, not necessarily basic, such that l(C) = 0 **and** lhl(C) = 0.' However, while the conditions l(C) > = 0 and lhl(C) > = 0 are sufficient when the upper bound is removed, they are certainly not necessary. The remainder of this paper goes on to discuss more general sufficient conditions.

4. CO-OPERATING CYCLES

The new conditions can be illustrated in a simple example obtained by modifying the grammar given above:

S → a**SCC**

S → T

T → **C**Tbb

T → d.

The terminal strings are $a^m\mathbf{C}^n d(bb)^n (\mathbf{CC})^m$, of which those with m = n are acceptable. It is interesting to compare the former strings $(a\mathbf{C})^n d(bb\mathbf{CC})^n$, from the previous grammar, with the new acceptable strings $a^n\mathbf{C}^n d(bb)^n (\mathbf{CC})^n$. In the former case the repeating groups were a**C** and bb**CC**, both of zero length, so that as n increased indefinitely the stack length remained bounded. However, the new strings begin with a^n, whose length increases indefinitely with n. Thus in the new grammar although the cycles cannot be combined to form a composed cycle, they nevertheless 'co-operate' together to maintain stack-boundedness below, at the cost of unboundedness above. This is despite the fact that neither cycle has both the properties lhl(C) > = 0 and l(C) > = 0.

Co-operating cycles arise when they are 'chained' together, as in the second rule of the above grammar. Another example could be

S → aa**SCCCC**

S → T

T → **C**Tbb

T → eUf

U → **C**Ugg

U → h

with strings $(aa)^m\,\mathbf{C}^n e\mathbf{C}^p h(gg)^p f(bb)^n (\mathbf{CCCC})^m$, which do not underflow the stack if m = n + p. Here the S and T cycles (rules 1 and 3) are chained together by the rule S → T, and the T and U cycles are linked by T → eUf.

5. CHAINS OF CO-OPERATING CYCLES

Investigated now is the question 'What conditions must be placed on a linear chain of co-operating cycles in order for it to yield an infinity of acceptable strings?' Certainly any such cycle chain has only a finite number of cycles in it (repeats excluded). This is because the graph G_R of any grammar is essentially finite, and the cycles discussed correspond to disjoint loops in G_R. Hence each cycle has different non-terminals of G in it. The general case is where the relevant parts of the grammar yield the relationships:

$$S \overset{*}{\Rightarrow} y_1 N_1 z_1$$
$$N_1 \Rightarrow u_1 N_1 v_1$$
$$N_1 \Rightarrow y_2 N_2 z_2$$
$$N_2 \Rightarrow u_2 N_2 v_2$$
$$N_2 \Rightarrow y_3 N_3 z_3$$
$$N_3 \Rightarrow u_3 N_3 v_3$$
$$\vdots$$
$$N_n \Rightarrow u_n N_n v_n$$
$$N_n \Rightarrow w.$$

Here the y, z, w, u and v symbols stand for strings of terminal symbols, and also, for brevity and without undue confusion, for the lengths of those strings. Now let the ith cycle be repeated x_i times. Then the strings generated are:

$$y_1 u_1^{x_1} y_2 u_2^{x_2} y_3 u_3^{x_3} \dots y_n u_n^{x_n} w v_n^{x_n} z_n v_{n-1}^{x_{n-1}} z_{n-1} \dots v_2^{x_2} z_2 v^{x_1} z_1.$$

Now let the desired lowest acceptable stack length be L (of course this is often zero or one). Consider what happens at all stages during the deposition of such a string on the stack. To do this it is only necessary to look at the stack length at certain critical points, as indicated below by the vertical bars:

$$y_1 \,|\, u_1^{x_1} \,|\, y_2 \,|\, u_2^{x_2} \,|\, \dots,$$

and so on. Hence are derived a set of inequalities which must all be satisfied:

$$y_1 \qquad >= L$$
$$y_1+u_1x_1 \qquad >= L$$
$$y_1+u_1x_1+y_2 \qquad >= L$$
$$y_1+u_1x_1+y_2+u_2x_2 \qquad >= L$$
$$\vdots$$
$$y_1+u_1x_1+y_2+u_2x_2+\ldots+\ldots+y_n+u_nx_n \qquad >= L$$
$$y_1+u_1x_1+y_2+u_2x_2+\ldots+\ldots+y_n+u_nx_n+w >= L$$
$$y_1+u_1x_1+y_2+u_2x_2+\ldots+\ldots +y_n+(u_n+y_n)x_n+w >= L$$
$$y_1+u_1x_1+y_2+u_2x_2+\ldots+\ldots +(y_n+z_n)+(u_n+v_n)x_n+w >= L$$
$$y_1+u_1x_1+\ldots+(u_{n-1}+v_{n-1})x_{n-1} +(y_n+z_n)+(u_n+v_n)x_n+w >= L$$
$$\vdots$$
$$(y_1+z_1)+(u_1+v_1)x_1+\ldots+(u_{n-1}+v_{n-1})x_{n-1} +(y_n+z_n)+(u_n+v_n)x_n+w >= L$$

Now let

$$p_1 = \max(L-y_1,\ L-(y_1+y_2))$$
$$p_2 = \max(L-(y_1+y_2),\ L-(y_1+y_2+y_3))$$
$$\vdots$$
$$p_i = \max\left(L-\sum_1^i y_r,\ L-\sum_1^{i+1} y_r\right), \quad \text{for} \quad i<n$$
$$\vdots$$

and

$$p_n = \max\left(L-\sum_1^n y_r,\ L-\sum_1^n y_r - w\right).$$

Also let

$$q_n = \max\left(L-\sum_1^n y_r - w,\ L-\sum_1^n y_r - w - z_n\right)$$
$$q_{n-1} = \max\left(L-\sum_1^n y_r - w - z_n,\ L-\sum_1^n y_r - w-(z_n+z_{n-1})\right)$$
$$\vdots$$

and

$$q_i = \max\left(L-\sum_1^n y_r - w - \sum_{i+1}^n z_r,\ L-\sum_1^n y_r - w - \sum_i^n z_r\right), \quad \text{for} \quad 1 =< i < n.$$

Then, ignoring $y_1 >= L$, an obvious pre-condition of any solution, the inequalities reduce to

$$u_1x_1 \qquad >= p_1$$
$$u_1x_1+u_2x_2 \qquad >= p_2$$
$$u_1x_1+u_2x_2+u_3x_3 \qquad >= p_3$$
$$\vdots$$
$$u_1x_1+u_2x_2+\ldots+u_ix_i \qquad >= p_i$$
$$\vdots$$
$$u_1x_1+u_2x_2+\ldots+u_ix_i+\ldots+u_{n-1}x_{n-1} \qquad >= p_{n-1}$$
$$u_1x_1+u_2x_2+\ldots+u_ix_i+\ldots+u_{n-1}x_{n-1}+u_nx_n >= p_n$$
$$u_1x_1+u_2x_2 +\ldots+u_1x_i+\ldots+u_{n-1}x_{n-1}+(u_n+v_n)x_n >= q_n$$
$$u_1x_1+u_2x_2 +\ldots+(u_{n-1}+v_{n-1})x_{n-1}+(u_n+v_n)x_n >= q_{n-1}$$
$$\vdots$$
$$u_1x_1+(u_2+v_2)x_2 +\ldots+(u_{n-1}+v_{n-1})x_{n-1}+(u_n+v_n)x_n >= q_2$$
$$(u_1+v_1)x_1+(u_2+v_2)x_2 +\ldots+(u_{n-1}+v_{n-1})x_{n-1}+(u_n+v_n)x_n >= q_1.$$

The u_i, v_i, p_i and q_i are integer constants derived from the grammar and the stack length, while the x_i are positive integer variables, the numbers of repetitions of the cycles involved. It is well known that, in any one such inequality, the equality option is only possible if the constant p_i or q_i is a multiple of the highest common factor of the coefficients of the variables. It is now assumed before further treatment that all the p_i and q_i have been incremented until this is true, without loss of generality.

6. TWO CO-OPERATING CYCLES

When investigating this system of inequalities, it is helpful to consider first the case where $n = 2$, partly because this turns out to be a special case, as will emerge later, and partly because it illustrates the general treatment to follow. For $n = 2$, the inequalities are:

$$u_1x_1 \qquad >= p_1, \tag{1}$$
$$u_1x_1+ \quad u_2x_2 >= p_2, \tag{2}$$
$$u_1x_1+(u_2+v_2)x_2 >= q_2, \tag{3}$$
$$(u_1+v_1)x_1+(u_2+v_2)x_2 >= q_1. \tag{4}$$

The co-operation of the two cycles is only being investigated because there is no cycle for which the length and the left-hand length are both greater than or equal to zero. Hence it may be seen at once that if $u_1 >= 0$ then $u_1+v_1 < 0$, and if $u_2+v_2 >= 0$ then $u_2 < 0$. Co-operation also implies that the infinity of solution-pairs $[x_1, x_2]$ that we seek must be such that both x_1 and x_2 are unbounded in them. Since x_1 may be indefinitely large, inequality (1) can only be satisfied if $u_1 >= 0$, so certainly $u_1+v_1 < 0$. The case $u_1 = 0$ can also be ruled out, for then $v_1 < 0$, and consideration of the four inequalities shows that the co-operation of the two cycles would then be more difficult than success of the second cycle on its own. Furthermore, from (4), u_1+v_1 and u_2+v_2 cannot both be negative, so that $u_2+v_2 > 0$. Thus it follows that $u_2 < 0$.

Summarising: $u_1 > 0$, $u_1+v_1 < 0$
$u_2 < 0$, $u_2+v_2 > 0$.

The ratio v_i/u_i is used frequently and is negative in almost every interesting case. Hence define $R_i = -v_i/u_i$. If $u_1 > 0$ and $u_1+v_1 < 0$ then $R_1 > 1$, and similarly if $u_2 < 0$ and $u_2+v_2 > 0$ then $R_2 > 1$ also. Thus whenever $R > 1$, a cycle cannot possibly generate an infinity of acceptable strings by itself.

Lemma 7

If $R_1 > 1$, and $R_2 > 1$, then there exists an infinity of solutions $[x_1, x_2]$ of the four inequalities in the pairs (u_1, v_1), (u_2, v_2) if and only if:

(i) $u_1 > 0, \quad u_2 < 0$

and either

(ii*a*) $R_1 < R_2$

or

(ii*b*) $R_1 = R_2$ and $p_2(R_2-1)+q_1 = < 0$.

(This second condition in (ii*b*) is to ensure there is at least one solution.)

Proof of necessity

(i) This was argued just before the theorem.
(ii) For any single solution $[x_1, x_2]$,

From (2) $$x_2 < = \frac{1}{u_2}(p_2-u_1 x_1).$$

From (4) $$x_2 > = \frac{1}{u_2+v_2}[q_1-(u_1+v_1)x_1].$$

Hence
$$\frac{1}{u_2}(p_2-u_1 x_1) > = \frac{1}{u_2+v_2}[q_1-(u_1+v_1)x_1],$$

from which
$$u_1 x_1(R_2-R_1) > = p_2(R_2-1)+q_1.$$

Thus if $R_1 = R_2$, then the condition $0 > = p_2(R_2-1)+q_1$ is proved necessary. However, if R_1 and R_2 are not equal and there are now an infinity of solutions, it follows that x_1 can be indefinitely large, so that the left-hand side $u_1 x_1(R_2-R_1)$ is unbounded, and therefore must be positive. Hence $R_1 < R_2$.

Proof of sufficiency

Condition (i) implies that $u_1+v_1 < 0$ and $u_2+v_2 > 0$. It is now shown that if either (ii*a*) or (ii*b*) is true then for x_1 chosen arbitrarily large it is possible to find x_2 which satisfies all the three inequalities in which it occurs.

From (2) $$x_2 < = \frac{1}{u_2}(p_2-u_1 x_1).$$

From (3) $$x_2 > = \frac{1}{u_2+v_2}(q_2-u_1 x_1).$$

From (4) $$x_2 > = \frac{1}{u_2+v_2}[q_1-(u_1+v_1)x_1].$$

Condition (ii*a*)

For large x_1, (3) is always true if (4) is true.
Choose $x_1 = k(u_2+v_2)$, where k is any natural number. Then (2) requires

$$x_2 < = \frac{p_2}{u_2}+ku_1(R_2-1)$$

and (4) requires

$$\frac{q_1}{u_2+v_2}+ku_1(R_1-1) = < x_2.$$

Then since $R_1 < R_2$, k can be chosen sufficiently large for all requirements to be met whatever the values of the constant terms.

Condition (ii*b*)

Let
$$R_1 = R_1 = R.$$
$$h = \text{hcf}(u_1, -u_2)$$
$$H = \text{hcf}[-(u_1+v_1),\ u_2+v_2]$$
$$= (R-1)h.$$

Then there exist integers j, k such that

$$p_2 = jh$$

and

$$q_1 = kH$$

(because the original p_2, q_1 have been increased until this is true, as discussed before).

The given condition

$$p_2(R-1)+q_1 = < 0$$

now reduces to $$k = < -j.$$

The original inequalities are now shown to be consistent and to hold for infinitely many pairs x_1, x_2 under this condition.

Let
$$u_1 = c_1 h$$
$$-u_2 = c_2 h,$$

so that c_1, c_2 are co-prime positive integers.

Inequality (2) now becomes

$$x_2 < = \frac{1}{c_2}(c_1 x_1 - j),$$

(4) becomes

$$\frac{1}{c_2}(c_1 x_1 + k) < = x_2$$

while as before (3) is always true if (4) is true.

Then the given condition ensures that the upper and lower bounds for x_2 are ordered correctly. It remains to choose x_1 so that x_2 can be an integer – just choose x_1 so that $c_1 x_1 \equiv j \pmod{c_2}$ – this is always possible since c_1 and c_2 are co-prime. Thus there is at least one solution. Since there is an infinity of x_1 with this property, there is also an infinity of valid pairs x_1, x_2.

Theorem 17

If two cycles C_1 and C_2 can be chained together and if $R_1 > 1$ and $R_2 > 1$ (so that neither cycle can yield an infinity of acceptable strings by itself), then they can together generate an infinity of acceptable strings if and only if:

(i) $u_1 > 0$ and $u_2 < 0$

and either

(ii*a*) $R_1 < R_2$

or

(ii*b*) $R_1 = R_2$, and one acceptable string is known to exist.

Proof

This is just the numerical Lemma 7 restated in terms of the grammar. (The strictly numerical version is used as a tool later.)

7. FINDING SINGLE-CYCLE INFINITIES

In order to place co-operating cycles in their context, the more general computational problem is now considered of examining a grammar (or equivalently a program text at compile-time), to see if an infinity of strings can be generated. This examination is simplest if the case of single-cycle infinities is eliminated first. These arise, of course, where there exists a cycle C such that lhl $(C) >= 0$ and $1(C) >= 0$.

Assume we have a group of basic cycles with left-hand and right-hand lengths (u_1, v_1), $(u_2, v_2), \ldots (u_i, v_i), \ldots (u_k, v_k)$ as above, but now with the property that they can be used together to compose new cycles rather than be disjoint. A solution is found at once if a basic cycle is such that $u >= 0$ and $u+v >= 0$. Otherwise, cycles such that $u <= 0$ and $u+v < 0$, or $u < 0$ and $u+v <= 0$ are never helpful, leaving those for which either $u_i > 0$, $u_i+v_i < 0$ (call this the i-type), or $u_j < 0$, $u_j+v_j > 0$ (the j-type). It is easy to show that one of each type of cycle can be successfully combined provided $R_i <= R_j$, by an argument similar to Theorem 17. It is also easy to show that composing a cycle from several of the basic i-type cycles produces another i-type cycle whose R value cannot be greater than or less than the extreme values for the basic i-type cycles – and analogously for the j-type cycles. Nothing is therefore gained by combining cycles in this way, and it is only necessary to inspect the basic cycles themselves when searching for i and j candidates such that $R_i <= R_j$.

8. FINDING DOUBLE-CYCLE INFINITIES

A pair of co-operating cycles is only sought when the procedure above fails to find a single cycle, whether basic or composed, with the right properties. Now sought is an i-type basic cycle which chains to (rather than 'can be combined with') a j-type basic cycle, and for which $R_i <= R_j$. This is a simple search which either succeeds or fails. If it fails the question arises: is it possible to find a longer chain of co-operating cycles which still produces an infinity of strings? Rather surprisingly no such longer chain can exist, and the proof of the result is now developed and finally stated in Theorem 18.

9. THE CO-OPERATING CHAIN OF LENGTH n

Now investigated is the set of inequalities for n pairs $(u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n)$.

Lemma 8

If $R_n > 1$, and the inequalities for the n pairs (u_1, v_1), $(u_2, v_2), \ldots, (u_n, v_n)$ have a solution $[x_1, x_2, \ldots, x_n]$, then the $n-1$ pairs

$$\left(u_1, \frac{v_1}{R_n}\right), \left(u_2, \frac{v_2}{R_n}\right), \ldots, \left(u_{n-1}, \frac{v_{n-1}}{R_n}\right)$$

yield inequalities of analogous form which have a solution $[x_1, x_2 \ldots, x_{n-1}]$. (Comment: In general, of course, v_i/R_n is not an integer, but the above is the easiest way to represent the new pairs of coefficients. The new inequalities can be trivially restored to the former integer style by multiplying each by $|v_n|$.)

Proof

Since $R_n > 1$, then u_n and u_n+v_n have opposite signs, and the p_n inequality can be combined with the $q_n, q_{n-1}, \ldots, q_1$ inequalities in turn to yield eventually, on eliminating x_n:

$$u_1 x_1 + u_2 x_2 + u_3 x_3 + \ldots \quad \ldots + u_{n-1} x_{n-1} >= \frac{1}{R_n}[q_n + (R_n - 1) p_n]$$

$$u_1 x_1 + u_2 x_2 + u_3 x_3 + \ldots + u_{n-2} x_{n-2} + \left(u_{n-1} + \frac{v_{n-1}}{R_n}\right) x_{n-1} >= \frac{1}{R_n}[q_{n-1} + (R_n - 1) p_n]$$

$$u_1 x_1 + u_2 x_2 + \ldots + \left(u_{n-2} + \frac{v_{n-2}}{R_n}\right) x_{n-2} + \left(u_{n-1} + \frac{v_{n-1}}{R_n}\right) x_{n-1} >= \frac{1}{R_n}[q_{n-2} + (R_n - 1) p_n]$$

$$\vdots$$

$$\left(u_1 + \frac{v_1}{R_n}\right) x_1 + \left(u_2 + \frac{v_2}{R_n}\right) x_2 + \ldots + \left(u_{n-1} + \frac{v_{n-1}}{R_n}\right) x_{n-1} >= \frac{1}{R_n}[q_1 + (R_n - 1) p_n].$$

Now let

$$P_i = p_i, \quad 1 =< i =< n-2,$$

$$P_{n-1} = \max\left(p_{n-1}, \frac{1}{R_n}[q_n + (R_n - 1) p_n]\right)$$

and

$$Q_i = \frac{1}{R_n}[q_i + (R_n - 1) p_n], \quad 1 =< i =< n-1.$$

Then it is established that the following set of inequalities also hold:

$$u_1 x_1 >= P_1$$

$$u_1 x_1 + u_2 x_2 >= P_2$$

$$\vdots$$

$$u_1x_1+u_2x_2+u_3x_3+\ldots+u_{n-2}x_{n-2} \qquad >= P_{n-2}$$

$$u_1x_1+u_2x_2+u_3x_3+\ldots+u_{n-2}x_{n-2}+u_{n-1}x_{n-1} >= P_{n-1}$$

$$u_1x_1+u_2x_2+u_3x_3+\ldots+u_{n-2}x_{n-2}+\left(u_{n-1}+\frac{v_{n-1}}{R_n}\right)x_{n-1} >= Q_{n-1}$$

$$u_1x_1+u_2x_2+\ldots+u_{n-3}x_{n-3}+\left(u_{n-2}+\frac{v_{n-2}}{R_n}\right)x_{n-2}+\left(u_{n-1}+\frac{v_{n-1}}{R_n}\right)x_{n-1} >= Q_{n-2}$$

$$\vdots$$

$$u_1x_1+\left(u_2+\frac{v_2}{R_n}\right)x_2+\ldots+\left(u_{n-2}+\frac{v_{n-2}}{R_n}\right)x_{n-2}+\left(u_{n-1}+\frac{v_{n-1}}{R_n}\right)x_{n-1} >= Q_2$$

$$\left(u_1+\frac{v_1}{R_n}\right)x_1+\left(u_2+\frac{v_2}{R_n}\right)x_2+\ldots+\left(u_{n-2}+\frac{v_{n-2}}{R_n}\right)x_{n-2}+\left(u_{n-1}+\frac{v_{n-1}}{R_n}\right)x_{n-1} >= Q_1$$

for the pairs

$$\left(u_1,\frac{v_1}{R_n}\right),\ \left(u_2,\frac{v_2}{R_n}\right),\ \ldots,\ \left(u_{n-1},\frac{v_{n-1}}{R_n}\right).$$

Lemma 9

If $n >= 2$, any set of inequalities (of the form described above for n co-operating cycles) which has an infinity of solutions $[x_1, x_2, \ldots, x_n]$ contains within that set of solutions an infinite subset in which either one or at most two of the x_i variables are unbounded.

Proof

The proof is by induction. The lemma certainly holds for $n = 2$. It is just necessary to prove the induction step. Let $n > 2$ and assume the lemma holds for $n-1$. Then there are three possibilities, labelled (a), (b) and (c).

(a) There exists i, $i =< n$, such that $u_i >= 0$, and $u_i+v_i >= 0$. Then as x_i increases it yields an infinity of solutions by itself. Q.E.D.

(b) There exists i, such that $u_i < 0$ and $u_i+v_i =< 0$, or such that $u_i =< 0$ and $u_i+v_i < 0$. Then every inequality involving u_i or u_i+v_i can be written:

[Left-hand side (less mention of x_i term]

$$>= (p \text{ or } q)-u_ix_i$$
$$>= (p \text{ or } q)$$

or similarly

[Left-hand side (less mention of x_i term]

$$>= (p \text{ or } q)-(u_i+v_i)x_i$$
$$>= (p \text{ or } q).$$

Thus this new set of inequalities in the $n-1$ variables $[x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]$ has an infinity of solutions and hence the lemma is proved by the induction assumption.

(c) all the $R_i > 1$ for all $i =< n$. Then $R_n > 1$, the condition for Lemma 8, and every solution $[x_1, \ldots, x_n]$ of the n inequalities leads to a solution $[x_1, \ldots, x_{n-1}]$ of the inequalities for the pairs

$$\left(u_1,\frac{v_1}{R_n}\right),\ \left(u_2,\frac{v_2}{R_n}\right),\ \ldots,\ \left(u_{n-1},\frac{v_{n-1}}{R_n}\right).$$

Since we are given that there are an infinity of solutions $[x_1, \ldots, x_n]$ there are also an infinity of solutions $[x_1, \ldots, x_{n-1}]$ of the derived inequalities. (The only way there might have been only a finite number of solutions $[x_1, \ldots, x_{n-1}]$ of the new set of inequalities would have been if all of $x_1, x_2, \ldots, x_{n-1}$ had only a bounded set of values in the original set of solutions $[x_1, \ldots, x_n]$ – but this would have meant that x_n alone was unbounded – and this is ruled out because $R_n > 1$.)

Consider the new set of inequalities obtained by eliminating x_n. Since there are only $n-1$ unknowns and an infinity of solutions, Lemma 9 itself can be applied. There are two cases, as follows.

(1) There exists $i <= n-1$ such that, as x_i alone increases, it generates an infinity of solutions by itself. In this case $u_i > 0$ and $u_i+v_i/R_n > 0$, so that $R_i < R_n$. ($u_i = 0$ is ruled out because at this stage it is known that $R_i > 1$.) Hence either:

$$(A) \quad u_n < 0 \quad \text{and} \quad u_n+v_n > 0$$

Lemma 7 now shows that the pairs (u_i, v_i) and (u_n, v_n) lead to an infinity of solutions in which just x_i and x_n are unbounded.

or $\qquad (B) \quad u_n > 0 \quad \text{and} \quad u_n+v_n < 0,$

while also

$$u_i > 0 \quad \text{and} \quad u_i+v_i < 0 \quad (\text{but yet } R_i < R_n).$$

It is now shown under these conditions that if $[x_1, x_2, \ldots, x_i, \ldots, x_n]$ is any solution, then so is $[x_1, x_2, \ldots, x_{i-1}, X_i, x_{i+1}, \ldots, x_{n-1}, 0]$ for some new integer X_i, and that the set of these new solutions is infinite. The lemma then follows from the inductive assumption. The argument begins with the inequalities involving x_i and/or x_n:

$$u_1x_1+u_2x_2+\ldots+u_ix_i \qquad >= p_i$$

$$u_1x_1+u_2x_2+\ldots+u_ix_i+u_{i+1}x_{i+1} \qquad >= p_{i+1}$$

$$\vdots$$

$$u_1x_1+u_2x_2+\ldots+u_ix_i+u_{i+1}x_{i+1}+u_{i+2}x_{i+2}+\ldots+u_nx_n >= p_n$$

$$u_1x_1+u_2x_2+\ldots+u_ix_i\ldots \qquad +(u_n+v_n)x_n >= q_n$$

$$\vdots$$

$$u_1x_1+u_2x_2+\ldots+u_ix_i+(u_{i+1}+v_{i+1})x_{i+1}+\ldots+(u_n+v_n)x_n >= q_{i+1}$$

$$u_1x_1+u_2x_2+\ldots+(u_i+v_i)x_i+(u_{i+1}+v_{i+1})x_{i+1}+\ldots+(u_n+v_n)x_n >= q_i$$

$$\vdots$$

$$(u_1+v_1)x_1+\ldots+(u_i+v_i)x_i+\ldots \qquad +\ldots+(u_n+v_n)x_n >= q_1.$$

It is proposed to increase x_i to X_i, and to reduce x_n to zero. This brings no difficulty in any of the inequalities until

$$u_1x_1+\ldots+u_ix_i+\ldots+u_{n-1}x_{n-1}+u_nx_n >= p_n$$

when $u_i(X_i-x_i) >= u_nx_n$ is required. Thereafter all inequalities remain satisfied as long as the coefficients of

x_i and x_n are respectively u_i and u_n+v_n, because $u_i > 0$ and $u_n+v_n < 0$. When later the coefficients are respectively u_i+v_i and u_n+v_n, reducing the value of x_n to zero increases the left-hand side by $-(u_n+v_n)x_n$, while changing the value of x_i increases the left-hand side by $(u_i+v_i)(X_i-x_i)$. Since the total increase must be non-negative

$$-(u_n+v_n)x_n+(u_i+v_i)(X_i-x_i) >= 0.$$

Hence

$$\frac{u_n+v_n}{u_i+v_i}\cdot x_n >= X_i-x_i >= \frac{u_n}{u_i}\cdot x_n,$$

which is possible only if

$$\frac{u_n+v_n}{u_i+v_i} >= \frac{u_n}{u_i},$$

which is equivalent to $R_i <= R_n$. This is the condition necessary for the removal of x_n to be justified. However the given condition was $R_i < R_n$, which guarantees the above, and shows that the X_i value sought does exist, certainly for sufficiently large x_n.

It remains to be shown that the $[x_1, \dots, x_{i-1}, X_i, \dots, x_{n-1}]$ solutions are an infinite set. Suppose not: since $x_i < X_i$, it then follows that the vectors $[x_1, \dots, x_{i-1}, x_i, \dots, x_{n-1}]$ constitute a finite set, so that the original set of vectors $[x_1, \dots, x_n]$ is then rendered infinite by the unboundedness of x_n alone. This is impossible since $u_n+v_n < 0$, and thus the $[x_1, \dots, x_{i-1}, X_i, x_{i+1}, \dots, x_{n-1}]$ are an infinite set.

(2) Although there exists no infinity of solutions as in (1) above, yet there exist $i, j =< n-1, i < j$, such that the pairs $(u_i, v_i/R_n)$ and $(u_j, v_j/R_n)$ co-operate together to yield an infinity of solutions, with only x_i and x_j being unbounded. Hence $R_i/R_n > 1$ and $R_j/R_n > 1$ (because it is known that neither pair yields an infinity by itself), so that by Lemma 7 $R_i/R_n =< R_j/R_n, u_i > 0$ and $u_j < 0$. Thus, since $R_n > 1$, it follows that $R_i =< R_j, u_i > 0$, and $u_j < 0$, conditions under which Lemma 7 can be invoked again to show that the original pairs (u_i, v_i) and (u_j, v_j) also co-operate to yield an infinity of solutions, as required. (In the case $R_i = R_j$ the additional condition is satisfied because it is known that at least one solution certainly exists.)

Theorem 18

If $n >= 2$, any chain of n co-operating cycles which yields an infinity of acceptable strings also has either one of its cycles such that this cycle yields an infinity of acceptable strings by itself, or two of its cycles which together form a co-operating pair.

Proof

The proof follows at once from lemma 9 by translating the wording of the theorem into the numeric style of the lemma. A separate lemma is necessary because its proof is inductive and involves the derivation of inequalities which have no correspondence in terms of cycles of the grammar.

10. 'TREES' OF CYCLES

The procedure for examining a particular program (i.e. the context-free grammar which corresponds to the program text), for an infinity of non-underflowing program runs was shown to be as follows.

(1) Determine whether there is any cycle in the grammar which generates an infinity of acceptable strings by itself. It is only necessary to examine each of the finite number of basic cycles in turn to see if $u >= 0$ *and* $u+v >= 0$.

(2) If this fails then examine the grammar for a pair of cycles chained together and which are able to generate the desired infinity by 'co-operating'. It is sufficient to examine pairs of co-operating basic cycles C_1 and C_2 for which $u_1 > 0$, $R_1 > 1$, $u_2 < 0$ and $R_2 > R_1$.

(3) If the above steps fail, it was shown that no longer chain of cycles could ever be found which would generate an acceptable infinity. The question arises: 'Is there any different, possibly more general, condition under which an acceptable infinity could still be generated?'

This article gives the answer 'Yes' to this question, and now introduces the notion of a tree rather than a chain of co-operating cycles. The whole argument is a generalization of the development of the theory of cycle-chains, and the techniques of proof are very similar. In consequence only very brief sketches of the proofs are given.

11. AN EXAMPLE CYCLE-TREE

Consider the grammar:

$$S \to aSCC$$
$$S \to TT$$
$$T \to CCTbbb$$
$$T \to d.$$

Then the R-value for the S-cycle is 2, and for the T-cycle 1.5. By expanding either the left-hand T or the right-hand T of rule (2) using the T-cycle of rule (3), the infinities of strings $a^m(CC)^n d(bbb)^n d(CC)^m$ and $a^m d(CC)^n d(bbb)^n (CC)^m$ are generated. However, these infinities underflow the stack, as may be seen by examining these particular cases, or by applying theorem 17, in which condition (ii*a*): $R_1 < R_2$ is contradicted. However, consider invoking the same number of repetitions of the T-cycle (rule (3)) for each non-terminal T of rule (2). The strings generated are then $a^m(CC)^n d(bbb)^n (CC)^n d(bbb)^n (CC)^m$. These cause no underflow provided that:

$$m-2n > 0,$$
$$m-2n+1+3n-2n > 0,$$
$$m-2n+1+3n-2n+1+3n-2m > 0.$$

These conditions are all satisfied if $m = 2n+1$. Thus an acceptable infinity is generated if both m and n increase indefinitely, but always retain the relationship $m = 2n+1$. The tree of three cycles 'co-operate together'.

12. SOME USEFUL TERMS

Cycle-tree. A finite tree of co-operating cycles which under suitable conditions generate an infinity of non-underflowing strings. The cycles are linked together by the rules of the grammar. (Of course a chain of cycles is a special case of a cycle-tree in which there is only one branch at every node.)

Base-cycle. The cycle at the base-node of the tree.

Cycle-branch. The sequence of cycles encountered on a path from the base-node to an extremity of a cycle-tree.
Leaf-cycle. A cycle at the end of a branch of a cycle-tree.
Depth of a cycle-tree. The maximum number of cycles on any branch of a cycle-tree.

13. FURTHER NOTATION

For a cycle-tree of depth two, let the pairs of lengths for the base-cycle be (u, v), and $(u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n)$ for the branches from left to right respectively.

14. THEOREMS AND LEMMAS

The formal results are now presented. Lemmas 10–12 and Theorems 19–20 are analogous respectively to Lemmas 7–9 and Theorems 17–18.

Lemma 10

Let there be a set of inequalities in $(u, v), (u_1, v_1), \ldots, (u_n, v_n)$ consistent with the form of a cycle-tree of depth two with n branches, $n >= 1$. If $R > 1$ and $R_i > 1$ for all i, $1 =< i =< n$, then there exists an infinity of solutions $[x, x_1, x_2, \ldots, x_n]$ if and only if

(i) $u > 0$, and $u_i < 0$, for all i

and either

(ii*a*) $R < R_1 R_2 \ldots R_n$

or

(ii*b*) $R = R_1 R_2 \ldots R_n$, and at least one solution is already known to exist.

Proof

By induction, and by eliminating x_n from the inequalities. The first step in the induction, when $n = 1$, is proved by Lemma 7.

The lemma may be restated in terms of co-operating cycles as follows.

Theorem 19

Let G contain a cycle-tree of depth two with n branches, $n >= 1$, with length-pairs (u, v), $(u_1, v_1), \ldots, (u_n, v_n)$. If $R > 1$ and $R_i > 1$ for all i, $1 =< i =< n$, then there exists an infinity of non-underflowing strings if and only if

(i) $u > 0$, and $u_i < 0$ for all i

and either

(ii*a*) $R < R_1 R_2 \ldots R_n$

or

(ii*b*) $R = R_1 R_2 \ldots R_n$, and at least one acceptable string is known to be generated by the cycle-tree.

Lemma 11

Consider a set of inequalities consistent with the existence of a cycle-tree of depth two at least. Let u^*, v^* be constants in the inequalities consistent with the existence of a leaf-cycle C^*, where $R^* = -v^*/u^* > 1$. Let x^* be the associated variable (in the model the number of cycle-repeats). Furthermore let there be an infinity of solutions of the inequalities. Then it is possible to derive another set of inequalities corresponding to just the same cycle-tree structure, except that:

(1) x^* has been eliminated (the chosen leaf-cycle C^* has been removed);

(2) any cycles between C^* and the base-node have a new R-value of R/R^*. (Any cycles on the left or right of C^* retain their original R-values.)

Proof

This is by elimination of x^*. The only way that the new set of inequalities need not have an infinity of solutions is if x^* had been the only variable to have an unbounded set of values in the original inequalities – but this is impossible because $R^* > 1$.

Lemma 12

Consider a set of inequalities consistent with the existence of a cycle-tree of any depth greater than two, with an infinity of solutions. Then either of the following must apply.

(1) There exists in the set a variable x_i and coefficients u_i, v_i such that $u_i >= 0$ and $u_i + v_i >= 0$, and there exists an infinity of solutions in which only x_i is unbounded.

(2) There exists in the set a subset of variables x, x_1, $x_2, \ldots, x_n$, for $n >= 1$, and coefficients (u, v), $(u_1, v_1), \ldots, (u_n, v_n)$ which are related by inequalities corresponding to a cycle-tree of depth two. This subset also has an infinity of solutions, and is such that $R =< R_1 R_2 \ldots R_n$, where also $R > 1$, and $R_i > 1$, for all i, $1 =< i =< n$.

Proof

The proof is on the lines of lemma 9 – an induction step is set up using a descent argument provided by lemma 11, and the using applications of lemma 12 itself. Descent stops at inequalities corresponding to a cycle-tree of depth two, and lemma 10 provides the needed result for $n = 2$.

Theorem 20

Let there be a cycle-tree of depth greater than two, which generates an infinity of non-underflowing strings. Then either of the following must apply.

(1) There exists a cycle in the tree for which $u >= 0$ and $u + v >= 0$, which generates an infinity of strings by itself.

(2) There exists a sub-tree of depth two for which, in the usual notation, $R > 1$, $R_i > 1$ for all i, $1 =< i =< n$, and $R =< R_1 R_2 \ldots R_n$, for some $n >= 1$. The cycles of the sub-tree co-operate together to generate an infinity of non-underflowing strings.

15. CONCLUSIONS

This concludes study of sufficient conditions under which an infinity of non-underflowing strings can be generated. A later article will show they are also necessary.

REFERENCES

1. Goodwin, D. T. Conditions for underflow and overflow of an arithmetic stack. *The Computer Journal* **20**, 56–62 (1977).
2. Goodwin, D. T. Partial non-underflow and non-overflow of an arithmetic stack. *The Computer Journal* **23**, 153–160 (1980).