3/14/78
revised 10/12/78

Programmable Calculator Algorithms

R. W. Newcomb

Introduction.

In general an algorithm is a set of procedures for carrying out an operation, such as a recipe for baking a cake. Since the structure of algorithms varies depending upon the context, as for example from cake baking to mathematical function evaluation, there are broad classes of algorithms [1][2][3]. Here we will consider algorithms in the context of programmable pocket calculators.

The Algorithm Concept.

A programmable calculator algorithm A is defined through a setting and transitionings within the setting, through the use of a program, as follows:

I. The Setting.

Let the following be finite sets

D = input domain, a set of vectors of modular numbers

R = output range, a set of vectors of modular numbers

S = state space, a set of vectors of modular numbers

P = program space = $\{P_0, P_1, \ldots, P_m\}$, P_i = ith (merged) key operation

then the setting $\mathcal{S}$ is

$$\mathcal{S} = D \oplus R \oplus S \oplus P, \qquad \oplus \text{ being the direct sum}$$

For a calculator we will take

$$D = N_d \oplus F \oplus \mathcal{O} \quad \text{and} \quad S = D \oplus N_p$$

where

N_d = space of data numbers enterable into data memory and display registers

N_p = space of numbers enterable into remaining program memory registers

F = space of flag settings

$\mathcal{O}$ = space of operation settings

We will take

$$R \subseteq N_d \cup \varphi, \qquad \varphi = \text{empty set, } \cup \text{ being set union and } \subset \text{ set containment}$$

Thus, if the calculator halts, the output is in a subset of data and display register readings. The state space is the full set of all register readings plus flag and operation conditions. The program space is like the set of keys which can be punched in writing a program and the index i on P_i can be considered as the numerical "key code" of the calculator. The algorithm setting is the set of all of these things.

II. The Transitioning

A) The Program

The transitioning for an algorithm is controlled by an n-step program $\mathcal{P}$, this latter being a finite sequenced set of n program steps p_i, $i = 0, \ldots, n-1$, taken from the program space P; that is,

$$\mathcal{P} = \text{program} = \{p_0, p_1, \ldots, p_{n-1}\}, \quad p_i \epsilon P$$

Here n is less than the maximum number of program memory registers; that is n is less than the cardinality of N_p.

B) The Transition Functions

There are three functions of interest all three depending upon the program. Thus let

$$\sigma = \text{state transition function} \qquad \sigma: S x \mathcal{P} \to S$$
$$\pi = \text{program transition function} \qquad \pi: S x \mathcal{P} \to \mathcal{P}$$
$$\rho = \text{output function} \qquad \rho: S x \mathcal{P} \to R$$

where

$$s_{i+1} = \sigma(s_i, p_{j_i}), \quad i = 0, 1, 2, \ldots; \quad j_i \epsilon \{0, 1, \ldots, n-1\}$$

$$p_{j_{i+1}} = \pi(s_i, p_{j_i})$$

$$r_i = \rho(s_i, p_{j_i})$$

with initial conditions

$$s_0 \epsilon D, \quad p_{j_0} = p_0$$

We will call a determination of σ, π, ρ, subject to s_0 and p_0, the <u>transitioning</u> $\mathcal{T}$.

The transitioning tells us that starting in the initial state s_0, given through the initial data readings, and at the initial program step p_0, we transition to the next state through the function σ and to the next program step through π. There are n program steps allowed, these being programmed in a sequence $p_0, p_1, \ldots, p_{n-1}$ [$n \leq 959$ for the TI SR-59]. These program steps are transitioned through in a sequence which depends upon the state and position in the program; hence π acts to permute the program steps leading to the permutation j_i on the indices. We can shorten the writing by defining

$$y = \begin{bmatrix} s \\ p \end{bmatrix}, \quad s \epsilon S, \; p \epsilon \mathcal{P}$$

$$A[y] = \begin{bmatrix} \sigma(s,p) \\ \pi(s,p) \end{bmatrix}$$

III. The Algorithm

An n-step algorithm A is the specification of the transitioning $\mathcal{J}$ through an n-step program within the setting $\mathcal{S}$.

A computation of the algorithm A is the sequenced pair

$$y_1, y_2, \ldots$$

$$r_1, r_2, \ldots$$

where, with initial conditions $y_0 = \begin{vmatrix} s_0 \\ p_0 \end{vmatrix}$

$$y_1 = A[y_0], \quad r_1 = \rho[y_1] = \rho(s_1, p_{j_1})$$

$$y_2 = A[y_1], \quad r_2 = \rho[y_2]$$

$$\vdots \qquad\qquad \vdots$$

If the sequence terminates, say at the kth step, then it is a terminating algorithm. The algorithm then calculates the function

$$f(x) = \rho[y_k], \quad x \epsilon N_d$$

where x is the projection of the initial state s_0 on the external data subset N_d of the state space. In this case $f(\cdot)$ must be a recursive function.

Generalization and Comments.

The above description is of a deterministic algorithm. If the transition is done through binary relations, rather than functions, the algorithm can be generalized to become nondeterministic. As we are usually just interested in $\rho[y_k]$ for the output, we can take $r_1, r_2, \ldots, r_{k-1}$ to be in the empty set in the computation. For digital filters we consider sequences of inputs $\{x_i\}$, $x_i \epsilon N_d$, and sequences of corresponding outputs, $\{f(x_i)\}$.

References

[1]. A. A. Markov, "Theory of Algorithms," published by the Israel Program for Scientific Translations, Jerusalem, 1962 (translated from Izdatel'stvo Akademii Nauk SSSR, Moskva-Leningrad, 1954)

[2]. J. Bruno and K. Steiglitz, "The Expression of Algorithms by Charts," in R. Rustin, Editor, "Algorithm Specification," Prentice-Hall, 1971, pp. 97-115.

[3]. J. E. Hopcroft and J. D. Ullman, "Formal Languages and Their Relation to Automata," Addison-Wesley, 1969, Section 1.2.

Example 1. Terminating algorithm for TI - 59

The algorithm calculates the n+1 Fibonacci type numbers satisfying

$$F_k = F_{k-1} + F_{k-2}, \quad k = 2, 3, \ldots, n+2, \quad n > 0,$$

with F_0 and F_1 prescribed.

We will use the minimum number 10 of data memory registers and set an error flag, number 8, to stop operation in case something goes wrong.

F: A space of 10-vectors for the TI-59, with entries 0 or 1, these corresponding to the 10 flags of the calculator. Here by the keyboard entries st flg 8 we place a 1 in position 8 (which will cause a halt on the occurrence of an error).

O: A space of 40-vectors for the TI-59 with entries 0 or 1. For this algorithm set all entries to be zero except the 17th, which is set as 1 op 17 on the keyboard, to partition registers to 10 data memory registers and 880 program memory registers.

N_d: A space of 12-vectors, the first 10 entries being the numbers in the 10 data memory registers and the next two corresponding to the, x, display register and the, t, test register. In this example we shall initially enter n in the t register and F_0 in data register 05, F_1 in data register 03. The program will enter k-2 in register 00 and use register 04 for F_{k-2}, register 05 for F_{k-1} and register 03 for F_k.

P: The program space P is the set of 100 merged key stroke operations $\{P_0, P_1, \ldots, P_{99}\}$ indexed by the key code (p. V-50 of TI-59 "Personal Programming" manual). For example $P_0 = 0$, P_{43} = RCL, P_{62} = Pgm Ind, P_{99} = Prt. Our program is $\rho = \{p_0, \ldots, p_{23}\}$ with $p_0 = P_{43}$ = RCL, $p_1 = P_5 = 5$, etc., as printed out in the following listing:

Meaning:			
previous F_{k-1}	000	43	RCL
	001	05	05
set = new F_{k-2}	002	42	STO
	003	04	04
previous F_k	004	43	RCL
	005	03	03
set = new F_{k-1}	006	42	STO
	007	05	05
	008	53	(
F_{k-1}	009	24	CE
+	010	85	+
F_{k-2}	011	43	RCL
	012	04	04
=	013	54	)
F_k	014	42	STO
	015	03	03
print F_k	016	99	PRT
recall k-2	017	43	RCL
	018	00	00
check is k-2 = n	019	67	EQ
if yes halt	020	92	RTN
if no increment k-2	021	69	OP
by 1	022	20	20
and start again	023	81	RST

Outputs during two cycles through the program for $F_0 = F_1 = 1$, $n = 5$.

r_i	p_{j_i}	
0.	RCL	$= p_0$
	5	
1.		
1.	STO	$= p_2$
	4	
1.		
1.	RCL	
	3	
1.		
1.	STO	
	5	
1.		
1.	(	
1.	+	
1.	RCL	
	4	
1.		
1.	)	
2.		
2.	STO	
	3	
2.		
2.	PRT	
2.	RCL	
	0	
0.		
0.	EQ	
	RTN	
0.	OP	$= p_{21}$
	20	
0.		
	RST	
0.	RCL	$= p_0 = p_{j_{24}}$
	5	
1.		
1.	STO	$= p_2 = p_{j_{26}}$
	4	
1.		
1.	RCL	
	3	
2.		
2.	STO	
	5	
2.		
2.	(	
2.	+	
2.	RCL	
	4	
1.		
1.	)	
3.		
3.	STO	
	3	
3.		
3.	PRT	
3.	RCL	
	0	
1.		
1.	EQ	
	RTN	
1.	OP	$= p_{21} = p_{j_{45}}$
	20	
1.		
	RST	
1.	RCL	$= p_0 = p_{j_{48}}$
	5	
2.		

Taking all but the "outputs" which are $F_2, F_3, \ldots, F_7$ as empty, we get the desired results.

2.
3.
5.
8.
13.
21.

Note that the modular class of numbers useable on the TI-59 at any calculation is the set of numbers of the form $\pm N \times 10^{\pm E}$ where the upper range of $N \leq 9999999999$ depends on $E \leq 99$ in use at the time.

TITLE #499 Fibonacci Numbers PAGE 1 OF 1

TI Programmable Program Record

PROGRAMMER ______________ DATE 10/14/78

Partitioning (Op 17) ______ Library Module ______ Printer ______ Cards 2

PROGRAM DESCRIPTION

Forms and prints:

for $n > 0$ $F_k = F_{k-1} + F_{k-2}$ $k = 2, \ldots, n$

for $n < 0$ $F_{k-2} = F_k - F_{k-1}$ $k = 1, 0, -1, \ldots, n+2$

USER INSTRUCTIONS

STEP	PROCEDURE	ENTER	PRESS			DISPLAY
	Enter $n \in R_{01}$					
	For $n > 0$, $F_0 \in R_{05}$, $F_1 \in R_{03}$,		A			
	For $n < 0$, $F_0 \in R_{04}$, $F_1 \in R_{05}$,		B			

examples:

$n > 0$

8.	N
2.	F0
1.	F1
3.	
4.	
7.	
11.	
18.	
29.	
47.	F_8

$n < 0$

-8.	N
1.	F1
2.	F0
-1.	
3.	
-4.	
7.	
-11.	
18.	
-29.	
47.	F_{-9}

USER DEFINED KEYS

A, B, C, D, E, A', B', C', D', E'

DATA REGISTERS (INV List)

0	$k-2$ $\quad$ $-k+1$	0	
1	n	1	
2		2	
3	F_1 $\quad$ F_k	3	
4	F_{k-2} $\quad$ F_0	4	
5	F_0 $\quad$ F_{k-1} $\quad$ F_1	5	
6	$n > 0$ $\quad$ $n < 0$	6	
7	↑ initial values ↑	7	
8		8	
9		9	

LABELS (Op 08)

INV lnx CE CLR x⇌t x²
√x 1/x STO RCL SUM yˣ
EE ( ) ÷ GTO ×
SBR − RST + R/S .
+/− = CLR INV log CP
tan Pgm P→R sin cos CMs
Exc Prd |x| Eng Fix Int
Deg Pause x=t Nop Op Rad
Lbl x≥t Σ+ x̄ Grad St flg
If flg D.MS π List Write Dsz
Adv Prt

FLAGS 0 1 2 3 4 5 6 7 8 9

TITLE *#499 Fibonacci Numbers* PAGE *1* OF *1*

PROGRAMMER ______ DATE ______

TI Programmable Coding Form

LOC	CODE	KEY	COMMENTS
000	76	LBL	
001	11	A	
002	03	3	
003	01	1	= N
004	69	OP	
005	04	04	
006	43	RCL	
007	01	01	n
008	69	OP	
009	06	06	
010	53	(	
011	24	CE	
012	75	−	
013	02	2	
014	54	)	
015	32	X⇄T	n − 2 ∈ t
016	02	2	
017	01	1	= F
018	00	0	
019	01	1	= 0
020	69	OP	
021	04	04	
022	43	RCL	
023	05	05	F_0
024	69	OP	
025	06	06	
026	02	2	
027	01	1	= F
028	00	0	
029	02	2	= 1
030	69	OP	
031	04	04	
032	43	RCL	
033	03	03	F_1
034	69	OP	
035	06	06	
α 036	43	RCL	
037	05	05	F_{k-1}
038	42	STO	set as
039	04	04	$F_{k-2} \in R_{04}$
040	43	RCL	
041	03	03	F_k
042	42	STO	set as
043	05	05	$F_{k-1} \in R_{05}$
044	53	(	
045	24	CE	F_{k-1}
046	85	+	+
047	43	RCL	
048	04	04	F_{k-2}
049	54	)	
050	42	STO	
051	03	03	$F_k \in R_{03}$
052	99	PRT	β
053	43	RCL	
054	00	00	k − 2

LOC	CODE	KEY	COMMENTS
055	67	EQ	is k − 2 = n − 2
056	92	RTN	if yes stop
057	69	OP	if not add
058	20	20	1 to k − 2
059	61	GTO	
060	00	00	} α
061	36	36	
2			
3			
4			
5			
6			
7			
8			
9	side 2		
240	76	LBL	
241	12	B	
242	03	3	
243	01	1	= N
244	69	OP	
245	04	04	
246	43	RCL	
247	01	01	n
248	69	OP	
249	06	06	
250	94	+/−	− n
251	53	(	
252	24	CE	
253	75	−	
254	01	1	
255	54	)	
256	32	X⇄T	− n − 1 ∈ t
257	02	2	= F
258	01	1	
259	00	0	
260	02	2	= 1
261	69	OP	
262	04	04	
263	43	RCL	
264	05	05	F_1
265	69	OP	
266	06	06	
267	02	2	= F
268	01	1	
269	00	0	= 0
270	01	1	
271	69	OP	
272	04	04	
273	43	RCL	
274	04	04	F_0
275	69	OP	
276	06	06	
277	43	RCL	
278	05	05	F_{k-1}
279	42	STO	set as

LOC	CODE	KEY	COMMENTS
280	03	03	$F_k \in R_{03}$
281	43	RCL	
282	04	04	F_{k-2}
283	42	STO	set as
284	05	05	$F_{k-1} \in R_{05}$
285	53	(	
286	24	CE	
287	94	+/−	$-F_{k-1}$
288	85	+	+
289	43	RCL	
290	03	03	F_k
291	54	)	
292	42	STO	
293	04	04	$F_{k-2} \in R_{04}$
294	99	PRT	
295	43	RCL	
296	00	00	−k + 1
297	67	EQ	is −k + 1 = −n − 1 (i.e. k = n)
298	92	RTN	if yes stop
299	69	OP	if not add
300	20	20	1 to −k − 1
301	61	GTO	
302	02	02	} β
303	77	77	
4			
5			
6			
7			
8			
9			
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			

MERGED CODES

62	Pgm	Ind	72	STO	Ind	83	GTO	Ind
63	Exc	Ind	73	RCL	Ind	84	Op	Ind
64	Prd	Ind	74	SUM	Ind	92	INV	SBR

TEXAS INSTRUMENTS INCORPORATED