5p

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
PROJECT MAC

Artificial Intelligence
Memo No. 32

On Efficient Ways of Evaluating Certain Recursive Functions

John McCarthy

The purpose of this memorandum is to illustrate a method for evaluating a recursive function when the same subexpression may occur many times during the evaluation and should be evaluated only once. An extreme example of this is the linear recursion

$$c_n = (n=0 \rightarrow a_0,\ n=1 \rightarrow a_1,\ T \rightarrow \alpha c_{n-1} + \beta c_{n-2})$$

If these equations are translated directly into LISP the evaluation of c_n will take approximately 2^{n-2} steps. Thus

$$c_5 = \alpha c_4 + \beta c_3 \quad \text{where}$$

$$c_4 = \alpha c_3 + \beta c_2$$

and the two c_3's are evaluated separately. Naturally, we can rewrite the recursion as

$$c_n = c(n,1,a_0,a_1)$$

where

$$c(n,m,a,b) = (n=0 \rightarrow a,\ m=n \rightarrow b;\ T \rightarrow c(n,m+1,b,\alpha b+\beta a))$$

However, I would like to consider a general method which works when we don't know which earlier values of the function will be required. Consider the problem of evaluating the number of partions of the number n, i.e. the number of ways n can be expressed as a sum. The partions of 5 are 5, 4+1, 3+2, 3+1+1, 2+2+1, 2+1+1+1, and 1+1+1+1+1, or more compactly 5, 41, 32, $2^2 1$, 21^3, 1^5.

The recursion is best accomplished by the aid of a function q_{mn} which is the number of ways m can be expressed as a sum each summand of which is no larger than n. Thus, $q_{55}=7$, $q_{54}=6$, $q_{53}=5$, $q_{52}=3$, $q_{51}=1$. We have the recursive relation

$$q_{mn} = (m{-}1 \vee n{-}1 \rightarrow 1,\ m \leq n \rightarrow q_{m,m-1},\ +\ 1T \rightarrow q_{m-n},\ n + q_{m,n-1})$$

Again, using this relation as a computation rule is inefficient in that certain q's will be evaluated many times. Therefore, we shall write equations for a procedure that keeps track of all q's that it has so far evaluated and will not evaluate any q more than once.

$$q_{mn} = \mathrm{val}[m;n;\ \mathrm{prob}[m;n;\mathrm{NIL}]]$$

$$\mathrm{val}[m;n;\mathrm{known}] = [\mathrm{eq}[\mathrm{caar}[\mathrm{known}];m] \wedge \mathrm{eq}[\mathrm{cadar}[\mathrm{known}];n] \rightarrow \mathrm{caddar}[\mathrm{known}];T \rightarrow \mathrm{val}[m;n;\mathrm{cdr}[\mathrm{known}]]]$$

$$\mathrm{prob}[m;n;\mathrm{known}] = [\mathrm{present}[m;n;\mathrm{known}] \rightarrow \mathrm{known};T \rightarrow \lambda[[v];\ \mathrm{cons}[\mathrm{list}[m;n;v];\mathrm{known}]][[m=1 \vee n=1 \vee m=0 \rightarrow 1;\ m \leq n \rightarrow 1+\mathrm{val}[m;n-1;\mathrm{prob}[m;n-1;\mathrm{known}];\ T \rightarrow \lambda[[\pi];\mathrm{val}[m-n;n;\pi] + \mathrm{val}[m;n-1;\pi]]\ [\mathrm{prob}[m;n-1;\mathrm{prob}[m=n;n;\mathrm{known}]]]]]]$$

---

present[m;n;known] = ∼null[known] ∧ [[eq[caar[known];m] ∧ eq[cadar[known];n]] ∨ present[m;n;cdr[known]]]

In these functions <u>known</u> represents a table of the q's that have already been found in the form

(($\underline{m}$, $\underline{n}$, $\underline{q}_{mn}$), ...)

<u>present[m;n;known]</u> is true if the value of $\underline{q}_{mn}$ is listed among <u>known</u>,

<u>val[m;n;known]</u> gives this value,

<u>prob[m;n;known]</u> gives a new list which includes $\underline{q}_{mn}$ and any other q's that arose in the course of evaluating $\underline{q}_{mn}$.

This process calculates exactly the q's that are needed. The major inefficiency of this as a LISP program comes from the linear scan used to determine whether q_{mn} has previously been computed. An associative memory procedure with hash addresses would relieve this inefficiency.

In the present case it is easy to calculate q_{mn} in a systematic way by the following ALGOLic procedure:

<u>for</u> k <u>from</u> 1 <u>to</u> m

<u>begin</u>

for ℓ from 1 to n

begin

$q(k,\ell)$ = <u>if</u> $(k=0 \vee k=1 \vee \ell=1)$ <u>then</u> 1 <u>else if</u> $k \leq \ell$ <u>then</u> $1+q(k,\ell-1)$

otherwise q $(k,\ell-1) + q(k-\ell,\ell)$

end

end

It is not clear how many of the q's evaluated are unnecessary. Certainly some of them are.

In the above algorithm one might be worried about the storage occupied by the table of values. In this case one might decide not to store the values of the q's that were very easily evaluated on the grounds that it would not take too long to evaluate them each time they turn up.

Keeping track of what is known in this manner may have applications in artificial intelligence also.

Problem for the student:

Write a general procedure that will transform any LISP recursive calculation into one that is guaranteed to evaluate the function no more than once for any argument. (10 points)

Prove by recursion induction or otherwise that the new function is always equivalent to the old. (50 points)

CS-TR Scanning Project
Document Control Form

Date : 11 / 30 / 95

Report # AIM-32

Each of the following should be identified by a checkmark:
Originating Department:

- [x] Artificial Intellegence Laboratory (AI)
- [ ] Laboratory for Computer Science (LCS)

Document Type:

- [ ] Technical Report (TR)
- [x] Technical Memo (TM)
- [ ] Other: ____________

Document Information

Number of pages: 5 (9-images)
Not to include DOD forms, printer intstructions, etc... original pages only.

Originals are:
- [x] Single-sided or
- [ ] Double-sided

Intended to be printed as :
- [x] Single-sided or
- [ ] Double-sided

Print type:
- [x] Typewriter
- [ ] Offset Press
- [ ] Laser Print
- [ ] InkJet Printer
- [ ] Unknown
- [ ] Other: ____________

Check each if included with document:
- [ ] DOD Form
- [ ] Funding Agent Form
- [ ] Cover Page
- [ ] Spine
- [ ] Printers Notes
- [ ] Photo negatives
- [ ] Other: ____________

Page Data:

Blank Pages(by page number): ____________

Photographs/Tonal Material (by page number): ____________

Other (note description/page number):

Description : Page Number:
IMAGE MAP: (1-5) UN#'ED TITLE PAGE, 1-4
(6-9) SCANCONTROL, TRGT'S (3)

Scanning Agent Signoff:
Date Received: 11 / 30 / 95 Date Scanned: 12 / 12 / 95 Date Returned: 12 / 14 / 95

Scanning Agent Signature: Michael W. Cook

Rev 9/94 DS/LCS Document Control Form cstrform.vsd

Scanning Agent Identification Target

Scanning of this document was supported in part by the **Corporation for National Research Initiatives,** using funds from the **Advanced Research Projects Agency** of the **United states Government** under Grant: **MDA972-92-J1029.**

The scanning agent for this project was the **Document Services** department of the **M.I.T Libraries.** Technical support for this project was also provided by the **M.I.T. Laboratory for Computer Sciences.**