1Q-4 Linked Data Structures の記憶管理

前川 博俊、安田 弘幸、沢田 佳明、福田 譲治
ソニー 総合研究所

1. はじめに

記号処理の分野において、ポインタによってリンクされたデータ（以下リストデータ）は不可欠である。我々は、リストデータをメモリ上にそのデータの性質を活かして表現し、仮想記憶の処理とガーベジコレクション（GC）を効率よく実行できるデータ管理方法 Structure Oriented Storage Organization (SOSO) を考案した[1]。

2. リストデータとその管理

リストデータはポインタによってリンクされ、メモリ上で動的状態で局所性の少ないことをひとつの特徴とする。そのようなデータを、例えばページング方式[2]によって仮想記憶化した場合 (図-1(a))、そのデータの性質上、仮想化の処理及び仮想空間上での GC において、二次記憶のアクセス頻度が高くなり、全体の処理効率が低くなる傾向にある。ページングや GC の方法を工夫しても[3-6]、この処理効率を大きく向上させることは難しい。

リストデータは、従来、各データセルが全アドレス空間内に割り付けられているが、そのリンクに従ってアクセスできる限り、各ポインタはメモリ空間上に一意に表現される必要はない。我々は、この性質を活かして、リストデータをメモリ上に表現し、そのリンク情報によって仮想記憶化する記憶管理方式 SOSO を考案した (図-1(b))。

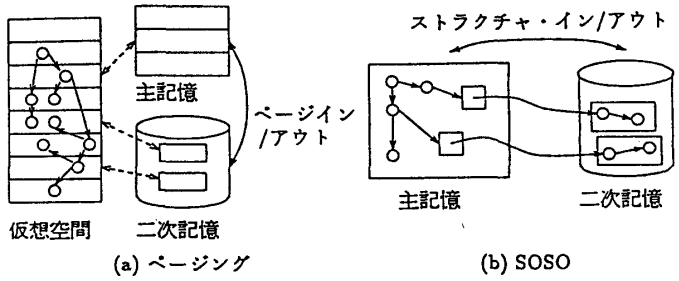

図-1 ページングとSOSOによる仮想記憶管理

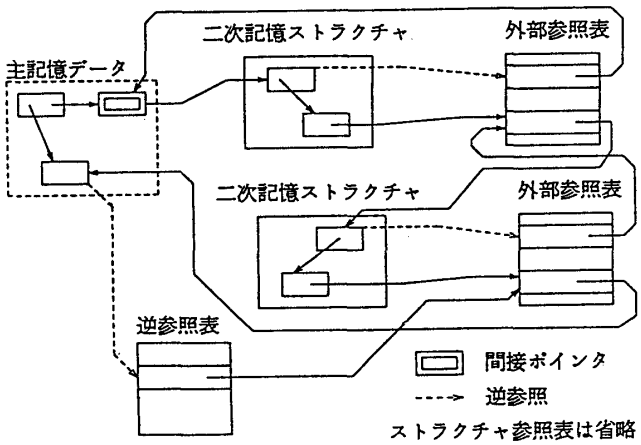

図-2 主記憶二次記憶間、二次記憶ストラクチャ間の参照管理

3. Structure Oriented Storage Organization(SOSO)

3.1. データ表現

主記憶上のポインタは、主記憶空間のアドレスで表現する。主記憶上で二次記憶上のデータは「間接ポインタ」によって参照する (図-2)。

主記憶と二次記憶間で転送するデータの単位を、ストラクチャと呼ぶ。二次記憶上のデータは、ストラクチャ単位に管理する。ストラクチャ内のポインタは、そのストラクチャ内のアドレスで表現する。

二次記憶ストラクチャ間、及び、二次記憶から主記憶に対する参照、被参照は、仮想化の処理の際の二次記憶アクセスを少なくするため、次の参照表で管理する (図-2)。

外部参照表： 二次記憶ストラクチャデータのストラクチャ外部に対する参照、被参照の管理。ストラクチャ毎に持つ。

逆参照表： 主記憶データの二次記憶に対する被参照の管理。

ストラクチャ参照表：
外部参照表の管理。外部参照表のコンパクション、および後述する参照カウントに使用する。

SOSOにおいて、各ポインタは、全仮想空間をアドレスするものである必要はない。SOSOでは、ここに述べた参照表を主記憶やメモリ制御部に持つ必要があるが、全空間

Memory Management for Linked Data Structures
Hirotoshi MAEGAWA,Hiroyuki YASUDA,Yoshiaki SAWADA and Goerge FUKUDA
Corporate Research Laboratories, Sony Corporation

にポインタを割り付ける従来の方式に比べて、ポインタのための記憶容量は、特に二次記憶において減少する。また、実空間、仮想空間ともに、独立にアドレス空間を拡張可能である。

3.2. 仮想記憶管理

二次記憶上のストラクチャデータを主記憶に転送することをストラクチャ・イン、実記憶上でストラクチャを抽出してそれを二次記憶に転送することをストラクチャ・アウトと呼ぶ。

ストラクチャ・アウトするデータは、例えば、制御の遠いところでの、アクセスや制御の退避環境を、被参照が少なくなるように抽出して得られる。

(1)ストラクチャ・インの処理

主記憶上で間接ポインタをアクセスしたとき、それが参照する二次記憶上のストラクチャを主記憶に転送する。その際、二次記憶上のストラクチャ表現を主記憶上のデータ表現へ変換する。転送したストラクチャ内に他の二次記憶ストラクチャへのポインタがあった場合、それは間接ポインタによって参照する。

ストラクチャ・インするデータが間接ポインタから参照されていたとき、その間接ポインタをなくし、直接参照に変換する。他の二次記憶ストラクチャから参照されていたとき、逆参照表によって、その参照元を主記憶データへの参照に書き換える。

(2)ストラクチャ・アウトの処理

ストラクチャ・アウトする主記憶上のデータは、二次記憶上のストラクチャ表現に変換し、転送する。そのストラクチャに主記憶上の他のデータセルから参照のあったとき、その被参照データセルを二次記憶上のデータへの間接ポインタとして残す。二次記憶上のデータから参照のあったときは逆参照表によって、その外部参照表を書き換える。

SOSOにおける仮想記憶のためのデータ転送に伴う処理は、例えば、ページング方式によるものに比べて複雑となるが、ストラクチャ単位に仮想記憶管理することによって、二次記憶装置へのアクセス頻度が減少することを期待できる。我々は、このデータ転送効率について、SOSO とページング方式を比較するため、 Lisp 処理系上でシミュレーションを行なった[1]。それによって、 SOSO はページング方式よりも、リストデータの仮想記憶管理において、効率の良い性質を持つことが確認できた。

3.3. GC

GC は、主記憶上ではデータセル単位に、二次記憶上ではストラクチャ単位に行なう。従って、実時間GCにおいても、一括GCにおいても、二次記憶を実際にアクセスする必要がなく、GCの高速実行を実現できる。

実時間 GC 及び仮想化のための間接ポインタの処理のため、主記憶上のデータセルは参照カウント (またはビット) を持つ必要があるが、これは主記憶上でのみ管理すれば良いので、メモリ制御部において効率の良い処理をすることが可能である。二次記憶ストラクチャの参照カウントは、ストラクチャ参照表で管理する。

4. おわりに

リストデータを、そのストラクチャに従ってメモリ上に表現し、仮想記憶及び GC を効率良く実現できるデータ管理方式 SOSO を報告した。特に GC は、二次記憶をアクセスすることなく実行することが可能で、リスト処理全体の処理能力の向上を期待できる。

我々は、 SOSO を用いた記号処理システムの研究、開発を行なっているが、 SOSO は既存の多くのシステムにおいても、メモリ制御を変更することによって組み込むことが可能である。また、メモリの物理的空間を分けることなどにより、ページング方式との併用も可能である。

謝辞

日頃から御指導いただいている、大阪大学　安井先生、京都大学　柴山先生、青山学院大学　井田先生、並びに、当社総合研究所　宮岡所長、総研ＡＶ研究所　吉田所長に深謝します。

【参考文献】

[1] 前川 他: Poniter-Linked Dataにおける仮想記憶管理の一手法,情報処理学会研究会資料, SYM50-1(1989).

[2] Harvey M. Deitel: An Introduction to Operating Systems, Addison-Wesley Publishing Company, pp.179-243 (1984).

[3] David L. Andre: Paging in Lisp Programs, The Faculty of Graduate School of the University of Meryland (1986).

[4] David A. Moon: Garbage Collection in a Large Lisp System, Conference Record of the 1984 ACM Symposium on Lisp and Functional Programming, pp. 235-246 (1984).

[5] David Ungar: Generation Scavenging: A Non-disruptive High Performance Storage Reclamation Algorithm, Proceedings of Software Engineering Symposium on Practical Software Development Environments, pp. 157-167 (1984).

[6] Henry Lieberman and Carl Hewitt: A Real-Time Garbage Collection Based on the Lifetimes of Objects, Communications of the ACM, 26, 6, pp. 419-429 (1983).