

4 A C - 1

永続分散共有メモリ機能を提供する データベースサーバ「わかし」 の64ビットアーキテクチャ上での実現*

城崎 徹 金子 邦彦 牧之内 顕文

九州大学大学院システム情報科学研究科知能システム学専攻†

1 はじめに

現在、コンピュータアプリケーションはより複雑になり、それに伴い巨大化の傾向にある。とくにそれはマルチメディアの分野で顕著で、MPEGやAudioなど、ギガ単位のファイルなどを扱うようになりつつある。

現在 Solaris 上(32ビットアーキテクチャ)で動作している、永続分散共有メモリ機能をもつデータベースサーバ「わかし」を64ビットアーキテクチャであるDEC社製 Digital UNIX 4.0A に移植した。

2 データベースサーバ「わかし」

「わかし」とは、我々の研究室で開発してきた「出世魚」というプロジェクトの中でベースとなるサーバを司る部分である。この出世魚は、

- 「大容量主記憶、高速ネットワークを利用してより高い性能を目指す」
- 「マルチメディアなどの応用を容易に開発することが可能な環境を提供する」

ということを念頭に置いて開発が進められてきた。「わかし」の機能は「分散共有仮想メモリ方式によるストレージ管理」である。つまり、ネットワークで相互に接続されたプロセッサ群において、データベースファイルが、クライアントの存在する遠隔サイトの仮想空間へ写像される。そして、1度遠隔サイトへデータが転送されるとそのデータに関する処理の多くが他のサイトに分散される。これは、そうすることによってサーバの仕事を複数のサイトに分散することができ、サーバボトルネックを解消することができるということを意味する。

* Persistent Distributed Shared Memory Server "WAKASHI" on 64-bit Operating System

† Toru JOUZAKI, Kunihiro KANEKO, Akifumi MAKINOUCHI (Graduate School of Information Science and Electrical Engineering, Department of Intelligent Systems, Kyushu University)

3 64ビットOS

すでに製品として提供されている64ビットOSとしては、DEC社製の Digital UNIX(DEC OSF/1)とSGIの IRIX がある。この他にもIBMやSun、HP、SCO等がこれから出荷予定である。

3.1 利用可能な実メモリ量の増大

OSは、スワップ領域と呼ばれるディスク上の特定の領域を利用して、物理的には存在しないメモリの領域までもプログラムからアクセスできるようにしている。しかし、ディスクとのI/OはメモリとのI/Oに比べてはるかに遅いため、なるべく多くの実メモリを積んでいた方がプログラムを高速に実行出来る。64ビットUNIXでは、10Gバイトのような大容量なメモリを搭載可能になる。大容量メモリの低価格化が進めば、Gバイト単位のメモリを搭載することが可能になり、処理の高速化を図ることが出来る。

3.2 大容量ファイル

ファイル内のデータの位置はファイル・ポインタと呼ばれるポインタによって管理されている。32ビットUNIXで取り扱うことの出来るファイルの最大のサイズが2Gバイトである。一方、64ビットUNIXでは、64ビット符合付き整数をファイル・ポインタを使うことによって、理論上 2^{64} バイトまでのファイルを扱うことが可能となる。このように、64ビットUNIXでは64ビットファイル・ポインタを使用することによって、32ビットUNIXの限界である2Gバイトを越え、はるかに大きいファイルの作成および利用が可能にする。

4 有効性

4.1 オン・メモリ処理による高速

32ビットUNIXでは、ユーザの利用出来るメモリ空間は2Gバイトまでと制限されている。このため、2Gバイト以上のデータを扱う場合には、データを適当なサイズに分割して扱う必要があった。このような場合では、ディスクI/Oが処理のボトルネックとなるこ

とが多い。しかし、64 ビット OS では、使用可能なメモリ空間に 2 G バイトという制限がないために、大量なデータを一度に扱うことが可能になる。

4.2 マルチメディア・データ

マルチメディア・データとは、jpeg 等の画像データ、MPEG 等の動画データ、Audio 等の音声データのことであり、これらのデータは、通常非常に巨大であったり、複雑な処理が必要となったりする。マルチメディアでのデータ処理では、次のようなものが要求されると考えられる。

- 大量のデータの読み書きの高速な処理
- 大量のデータの圧縮・展開の高速な処理
- 高速なグラフィックス処理
- 高速なデータ通信

64 ビット UNIX では、十分なアドレス空間、高速なバイナリ演算が可能であり、また巨大ファイルを扱うことも可能であるので、これらのマルチメディア分野の要求を満たすことが出来る。

5 32 ビットから 64 ビットアーキテクチャへの移植

「わかし」サーバは C++ で書かれている。そこで、C++ アプリケーションの 64 ビット化の問題点などについて考察する。

32 ビット OS では、int が long と同じサイズになっていることが多い。それに対し、LP64 の 64 ビット・データをサポートしている 64 ビット OS では int は今まで通り 32 ビットに対し、long は 64 ビットである。そのため long から int への代入で桁落ちが生じる可能性がある。

ポインタは long と相互に代入可能である。しかし、32 ビット OS では int と long が明確に区別されていないため、ポインタと int との相互代入が行われていると、int は 32 ビットで同じだがポインタが 64 ビットになったため、問題となる。

しかし、実装の段階で `off_t` や `size_t` といった型宣言子を使用していれば移植の際に問題はないが、int を直接使っているような場合はこれらの変更が必要である。

5.1 「わかし」の移植

現在「わかし」サーバは、32 ビット OS である Solaris 2.5 上で動作している。これを 64 ビット OS、Digital UNIX 4.0A に移植を行う。どちらの OS も (SVR4) System V Release 4 をベースとしているので、OS の仕組みは良く似ている。移植の際に一番の問題となる

点は、64 ビット化に伴う型の変更である。上記のように、32 ビットアプリケーションを 64 ビットに移植するには問題が生じる可能性がある。「わかし」サーバでは、型宣言で型宣言子を使用せずに、int、long と直接使用している。そのため、単純に 32 ビット版「わかし」をコンパイルするのは、不可能である。また、OS レベルからアーキテクチャが異なるので、システムコールの戻り値等も一部、異なるために、単純な型の書き換え作業だけではない。

そこで、この問題に対応するために、以下の様な手順で行った。

1. 全ての変数の洗いだし、変更を必要とする変数の決定
2. 将来の拡張に対して、変更が容易になるように型宣言子を決定する
例: `int mid;` を `size_t mid;` に変更
3. 実際の変更を行い、32 ビット上でも 64 ビット上でもコンパイル可能にする
4. Digital UNIX 4.0A 機種依存の関数を書き換える

特に変更を要したのは、メモリ空間が広がったことによる、ページ数、ヒープサイズの増加に伴う変更。ポインタとアドレスへのオフセット値の変更が挙げられる。

この際、将来の拡張、及び LP64 とは異なる 64 ビット OS への移植等を考え、型宣言子を用いるようにした。

6 結論

64 ビット・システムの利用により、「処理の高速化」、「大規模のデータ処理が可能」というメリットがある。また、ATM 等により高速な通信を可能にすることで、データベースの分野、特にマルチメディア・データを分散環境下で扱う上で 64 ビット化することは、非常に有効であると考えられる。

よって、「わかし」サーバを 64 ビットアーキテクチャ上への移植は、大容量なマルチメディア・データの取り扱いが可能となる。

参考文献

- [1] 清兼 義弘 編著, "64 ビット UNIX & CDE", 共立出版株式会社, 1997
- [2] 64bit Computing Today,
<http://www.UNIX.digital.com/unix/64bit/index.html>
- [3] 64bitUNIX 技術解説,
<http://www.dec-j.co.jp/ic/unix/technical/64bit/nc970106/unix64.html>