

共有メモリの書き込み同期機構の高速化

3F-3

園田 俊浩 西口 直樹 柴田 清己 本田 文雄 竹林 知善
富士通研究所

1 はじめに

マルチプロセスをサポートするオペレーティングシステムでは、クライアントプロセスとサーバプロセス間のように複数のプロセスでデータを共有したい場合がある[1]。このような場合、オペレーティングシステムが提供する共有メモリのサービスを使ってデータを共有し、データ間の整合を保つために共有領域へのアクセス同期を各プロセスが行う。通常、あるプロセスが共有データを変更するときは、他のプロセスのアクセスを禁止する必要がある。共有領域の変更に関連する処理時間が大きい場合、他の共有領域を利用したいプロセスは共有領域へのアクセス処理寸前でウェイトし続けることになるが、変更処理中でもウェイトせずに変更以前の状態を参照したい状況がある。

そこで、共有領域が変更されたとき他のプロセスに対してその変更の反映が遅延可能な場合、共有領域を変更するプロセスは、共有領域のコピー（以後、一時領域と呼ぶ）を生成しこれを変更する。変更は一時領域に対して行われるので、他のプロセスは共有データを参照できる。変更処理を行っているプロセスは、変更が終了した時点で一時領域を共有領域に更新する。この方法によって、あるプロセスが共有領域を変更中でもその共有領域を参照することが可能となる。

しかし、共有領域のサイズが大きくなると、一時領域を生成するためのコピーと変更後の共有領域の更新の負荷が大きくなるという問題が生じる。

本システムでは、この問題を解決するために、一時領域の生成をオペレーティングシステムで行うことにより高速化を実現した。またオペレーティングシステムが、変更された情報を保持し、維持領域の変更を共有領域に反映する際、変更箇所を含む特定の領域だけをコピーすることにより、この処理の高速化も実現した。以降では一時領域の生成から、変更の更新までをオペレーティングシステムの動作を中心に述べる。

2 一時領域の生成

共有領域を変更するプロセスは、その旨をオペレーティングシステムに依頼する。オペレーティングシステム

は指定された共有領域のための一時領域を生成するが、コピーは書き込みが起きたときまで遅延するコピーオンライト[2]によって行う。このとき、オペレーティングシステムは、「生成した一時領域の変更が終了すると、変更の内容が共有領域に更新される」という情報をリンクとして保持しておく。図1にこのときの様子を示す。

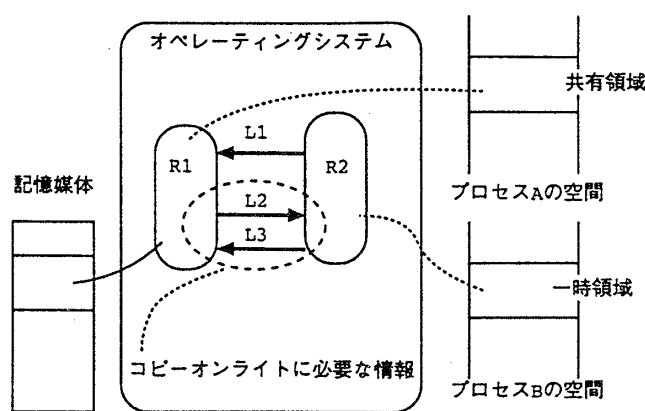


図1: 一時領域生成直後

図中の R1 は共有領域の管理情報、R2 は一時領域の管理情報、L2, L3 はコピーオンライトのために必要なリンク情報、L1 は一時領域と共有領域の関係を表すリンク情報である。プロセス B が共有領域を変更する旨をオペレーティングシステムに通知したときに、図1のような一時領域が生成される。

3 一時領域の変更

図1の時点で一時領域の保護属性を書き込み不可にしておく。その状態で一時領域が変更されると例外が発生し、オペレーティングシステムに通知される。オペレーティングシステムはその原因が一時領域への書き込みであることができて、この領域に対応する共有領域のデータをコピーし、一時領域にマップすることができる。

このときのコピーの量は MMU がプロテクト指定可能な物理メモリのページサイズとなる。例えば、Intel386 [3] かその継承 CPU [4] であれば 4 キロバイトとなる。

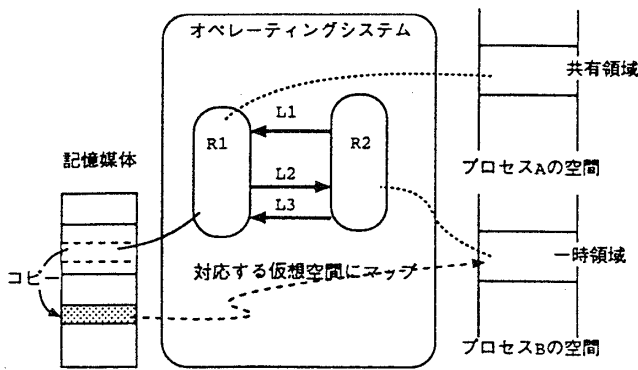


図 2: 一時領域へ書き込みが行われた場合

図 2 に一時領域の変更が発生した直後の様子を示す。R1, R2 は仮想領域情報の他にも物理ページのアロケート情報やマップ情報も保持している。

一時領域の変更された箇所に対応する物理ページだけが新しくアロケートされるため、物理ページのアロケート情報を R2 が保持しているということは、一時領域で変更された領域情報を保持していることに等しくなる。つまり、変更が起きた領域は R2 が保持している物理ページのアロケート情報から決定できることになる。

4 一時領域の更新

プロセス B は一時領域の変更が終了すると、その旨をオペレーティングシステムかプロセス A に通知する。プロセス A は更新可能なタイミングまで更新を遅延し、オペレーティングシステムに通知する。前述したように、オペレーティングシステムは変更した物理ページの情報も保持しているので、図 3 に示すように変更が起きた物理ページ P' のみを共有メモリにコピーできる。

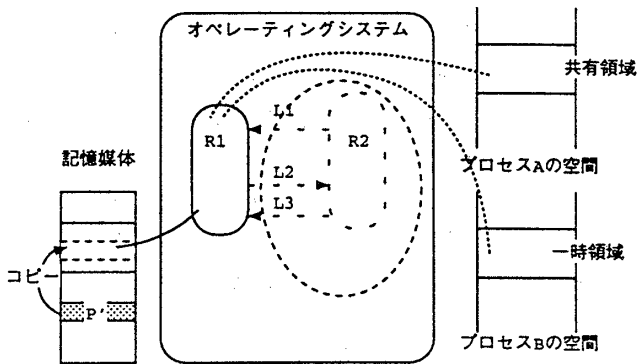


図 3: 一時領域の更新

最後に、一時領域用に作成した R2 や関連リンク情報 L1, L2, L3 を削除し、一時領域のマップを R1 の領域に変更する。

5 一時領域の効果

一時領域をオペレーティングシステムで管理することにより

- 一時領域の生成時に発生するコピー量を削減できる。
- 変更の更新時に発生するコピー量を削減できる。
- アプリケーションのバグを削減できる。

の利点がある。また、一時領域の生成に関してはコピーオンライトという従来のシステムをそのまま利用しているために、オペレーティングシステムの変更も少なく大幅な変更は必要ない。

6 おわりに

現行のシステムは同一マシン上にあるローカルな共有メモリを対象として、コピー量の削減に着目しているが、分散共有メモリのようにネットワークを解して共有メモリ間の一貫性を維持する必要があるケースに関しては、差分のみの更新メカニズムは通信データ量の削減につながる [5]。そのために、今後はネットワーク環境への応用に関しても考えていきたい。

参考文献

- [1] 西口直樹, 園田俊浩, 柴田清己, 本田文男, 竹林知善, “C++オブジェクトをプロセス間で共有するための管理機構について”, 情報処理学会 第 57 回全国大会, 1998
- [2] M.K.McKusick, K.Bostic, M.J.Karels, J.S.Quarterman, The Design and Implementation of the 4.4 BSD Operation System, Addison Wesley
- [3] J.H.Crawford, P.P.Gelsinger, 80386 プログラミング, 工学社,
- [4] Pentium プロセッサ アーキテクチャとプログラミング, インテル株式会社
- [5] Keleher, P., Dwarkadas, S., Cox, and Zwaenepoel, W., Tread-Marks: Distributed Shared Memory on Standard Workstations and Operating Systems, Proc. of the Winter 1994 USENIX Conference, pp.115-131(1994)