発表概要

C++におけるポインタベースのデータ構造のための Far Memory アロケータ

秀島 宇音[1]　佐藤 重幸[1]　田浦 健次朗[1]

2022年7月28日発表

Far Memory は，ネットワーク越しの他マシンのメモリをデータの置き場に加えつつ手元のメモリとの間でのデータ移動を自動的に行う技術であり，手元のメモリ資源を超えた規模のデータに対する多様な操作を，少ないプログラマ負担で実装できるようにする．これを C++などメモリへの低水準操作を許す言語に向けて提供するにあたっては，1) 独自の API を介さず通常のメモリ領域と同様に読み書きでき，2) かつ通常の領域と使い分けられることが望ましい．だが我々の知る限りこの両方を満たす実装は存在しない．そこで本発表では，メモリ空間内の部分空間に対して Far Memory 機能を付加し，そこに属するメモリ領域を専用のアロケータで確保できるようにする，アロケータとしての Far Memory 実装を提案する．これによりプログラマは，通常の領域と区別できる方法で Far Memory 領域を確保し，そこへの読み書きは通常の領域と同様に記述できるようになる．提案実装はそうした読み書きにページキャッシュを用意して対応しつつ，さらに局所性を意識したメモリ確保によって性能向上に寄与する．我々は提案実装の上にポインタベースの構造である平衡木を実装することを通じて，各々の特性を活かして性能を高めるプログラミングが実現されることと，提案実装が生む Far Memory 領域と通常の領域との高い相互運用性がプログラマの負担を軽減することを確認した．

Presentation Abstract

A Far-memory Allocator for Pointer-based Data Structures in C++

Takato Hideshima[1]　Shigeyuki Sato[1]　Kenjiro Taura[1]

Presented: July 28, 2022

Far memory is a technology that enables us to utilize the memory resources of remote machines as if they are local ones by transparently moving data through networks. It facilitates implementing various operations on larger-scale data than the memory resources of local machines. In C/C++, or languages supporting direct operations on memory, far-memory regions have two desiderata: 1) the same operations as normal regions are applicable to them; yet, 2) they support region-aware programming. Unfortunately, to the best of our knowledge, no implementation satisfies both of them. In this presentation, we propose an allocator implementation of far memory, which creates far-memory sub-regions within the memory space and provides a dedicated allocator for them. With this interface, programmers can perform region-aware allocation and direct read/write operations to the allocated far-memory regions. Our implementation handles such reads/writes with page cache, and furthermore, contributes to performance improvement by locality-aware allocation. Through implementing balanced trees atop the proposed allocator, we have confirmed that our allocator supports region-aware programming to improve performance and offers high productivity stemming from the interoperability with the existing C/C++ code.

This is the abstract of an unrefereed presentation, and it should not preclude subsequent publication.

[1] 東京大学情報理工学系研究科
Graduate School of Information Science and Technology, The University of Tokyo, Bunkyo, Tokyo 113–8656, Japan