針對可擴展性的並行資料結構設計與實作

The Design and Implementation for Scalable and Concurrent Data Structures

指導教授：陳奇業、黃敬群
專題成員：吳昱
開發工具：GCC、valgrind、gnuplot
測試環境：Linux Ubuntu 20.04

一、 簡介

研究動機：

現代處理器多為多核心架構，如何撰寫平行/並行程式以善用這樣的架構也成為了很重要的議題。與一般的程式不同，平行/並行程式需要考慮許多議題，最基本的如共享資料的正確性。更進階的如程式的可擴展性(scalability)。這個專題主要探討多核心環境中，提高 scalability 的手段。

研究內容：

一、lock-free 程式設計

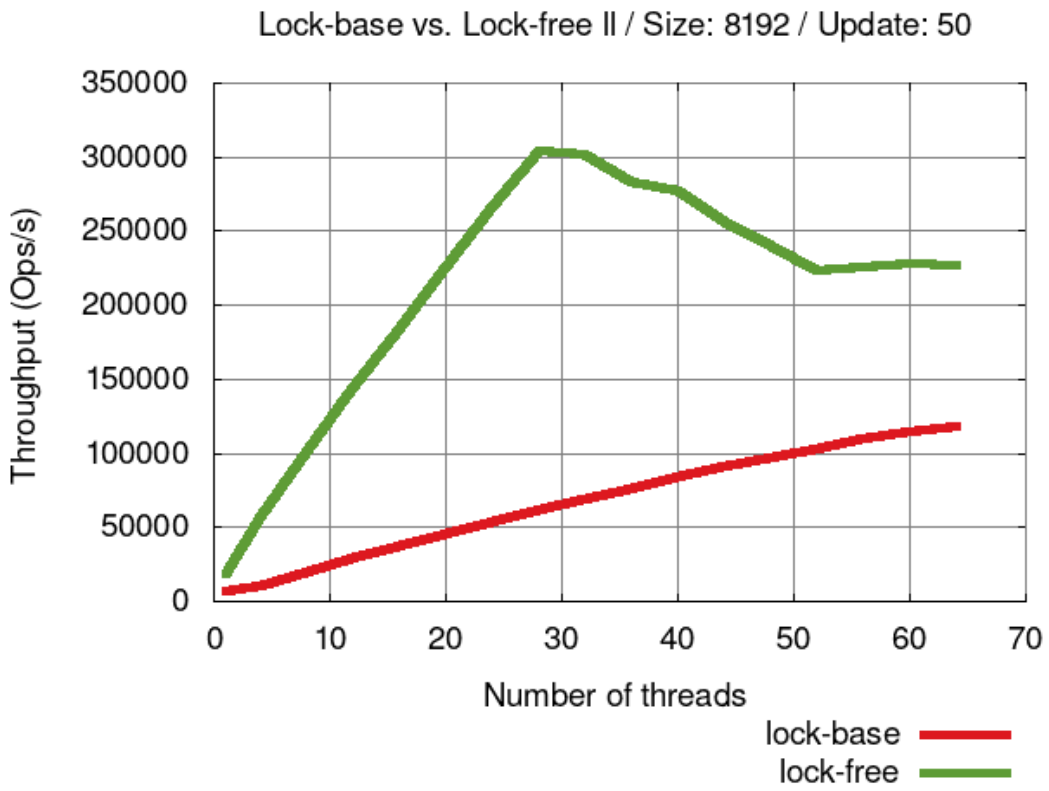
為了確保並行程式在讀/寫同一共享資料時不會發生錯誤，一種常見的程式設計是利用 lock 保證不同的程式不會同時進入 critical section。但是當並行程式的數量提高時，程式間對於 lock 的競爭常常會造成效能的瓶頸，並對可擴展性(scalability)造成嚴重的負擔。

因此，這個專題希望可以探討 lock-free 的程式設計對於可擴展性的提升。主要針對 singly linked list 這個資料結構實作 lock-base 、 lock-free 兩種不同的版本，並比較多核心的環境中兩者的效能表現。包括資料吞吐量(throughput)以及可擴展性(scalability)。

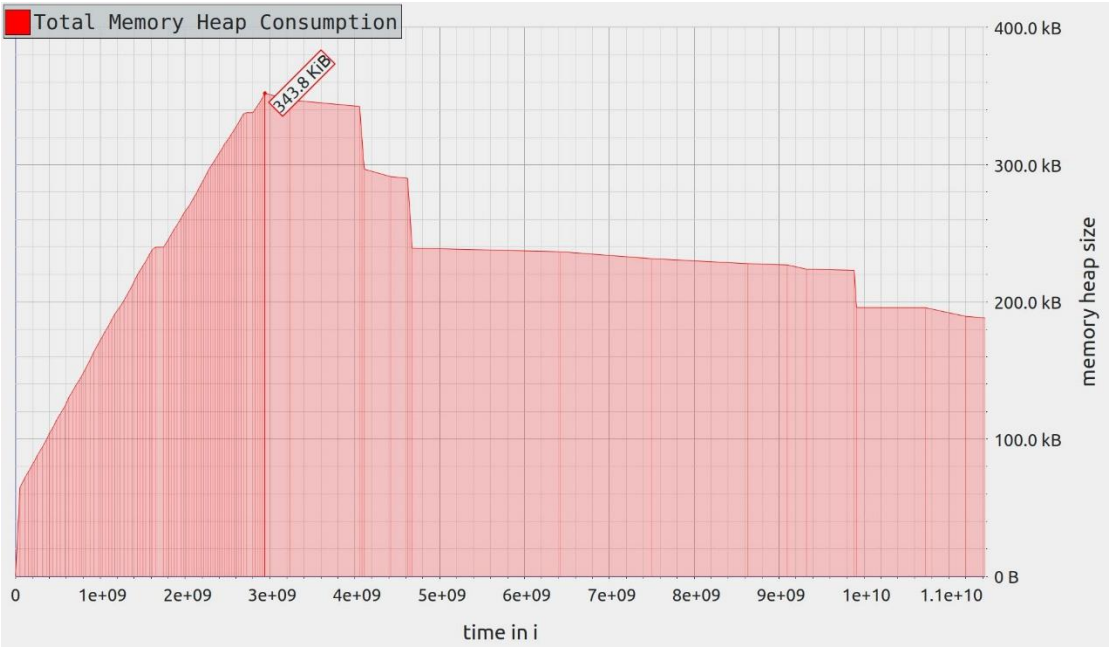
二、Memory reclamation

不同於 lock-base 的程式， lock-free 的程式在回收動態記憶體上更加困難，主要是因為待釋放的記憶體有可能正在被其他程式使用中，因此要更加小心。這個專題嘗試引入 hazard pointers 的機制來解決此問題。

二、測試結果：



上圖中，Lock-free linked list 的 throughput 高於 lock-base 的版本。



上圖中，使用的 hazard pointers 後， heap 的大小限制在 400 KB 以內。