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

研究動機

- 在多核心環境中，可擴展性(scalability)是很重要的議題
- 針對提升 scalability 的手段進行探討與實作

研究內容 Lock-free linked list

- 主要針對 singly linked list
- 基於論文 Lock-Free Linked lists and Skip Lists 進行 lock-free 實作
- 透過實驗比較 lock-base 以及 lock-free 版本的效能差異

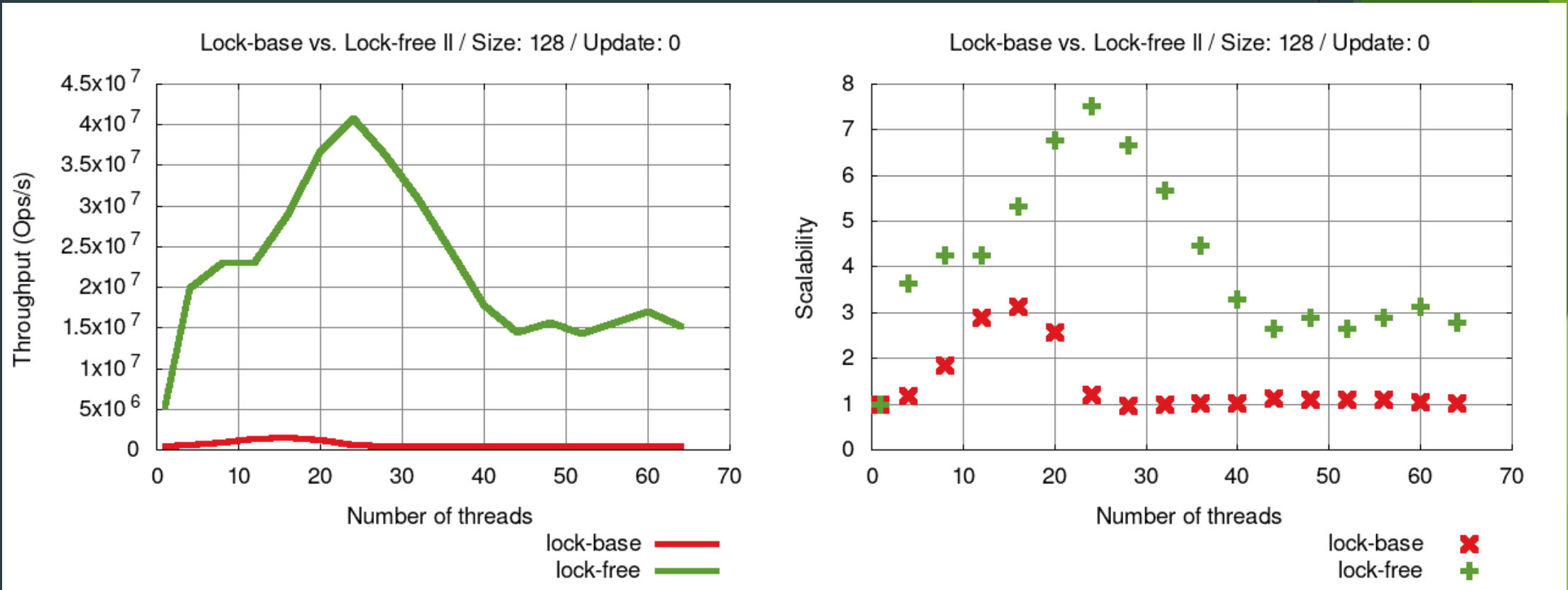
實驗環境

- Model name: ThunderX2
- Vendor : Cavium (2017年被 Marvell 併購)
- Arm, 4-way SMT
- 2 NUMA nodes
- 224 cores

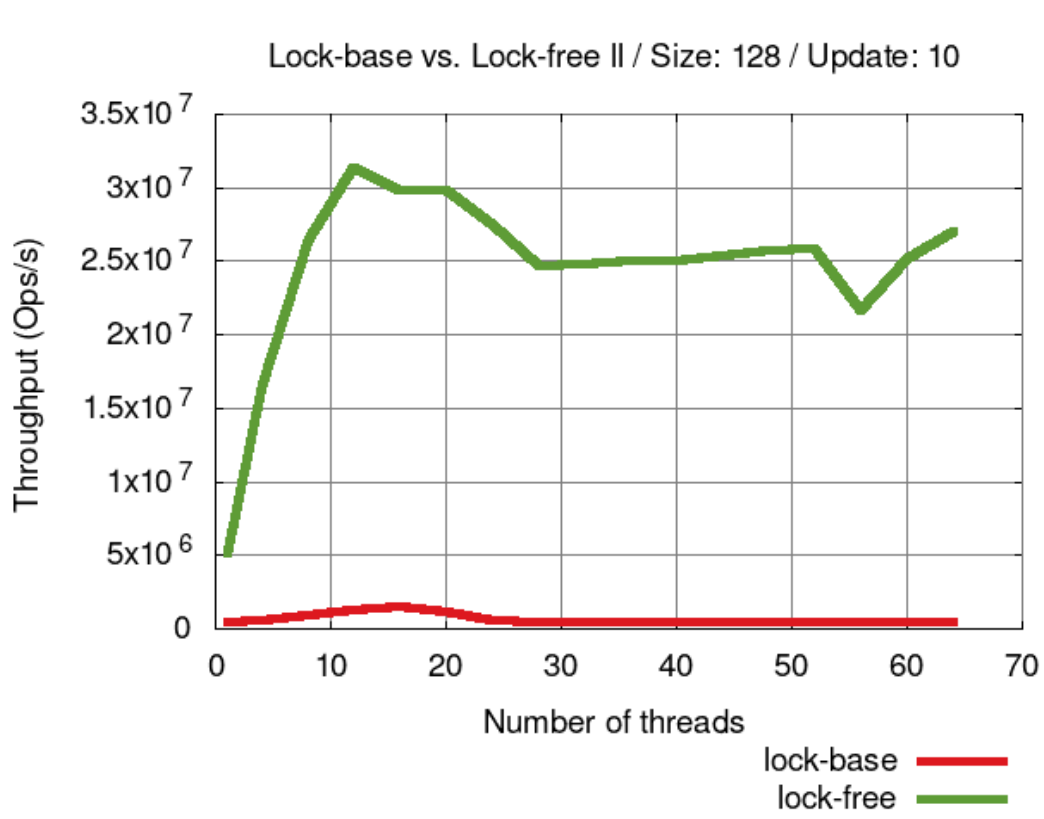
實驗參數

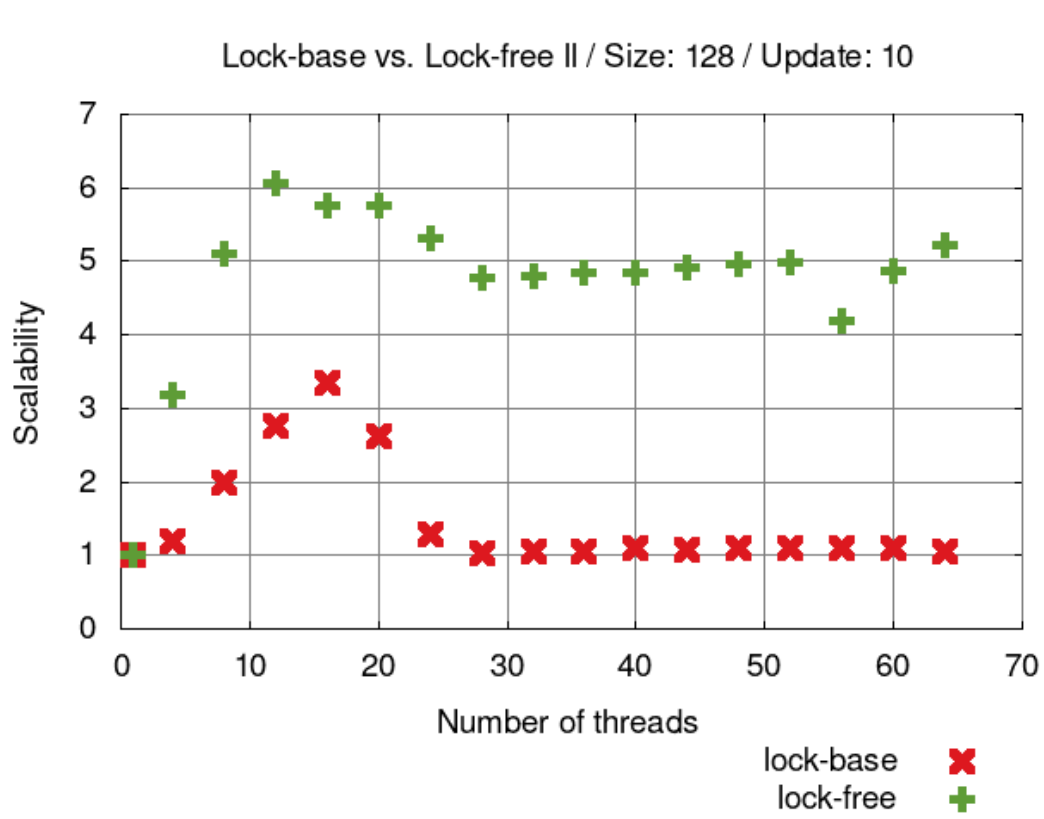
- Thread number: 1~64
- Linked list length: 128、1024、8192
- Update rate: 0%、10%、50%
- 每個執行緒都執行在固定的核，不會更換(thread affinity)

實驗結果

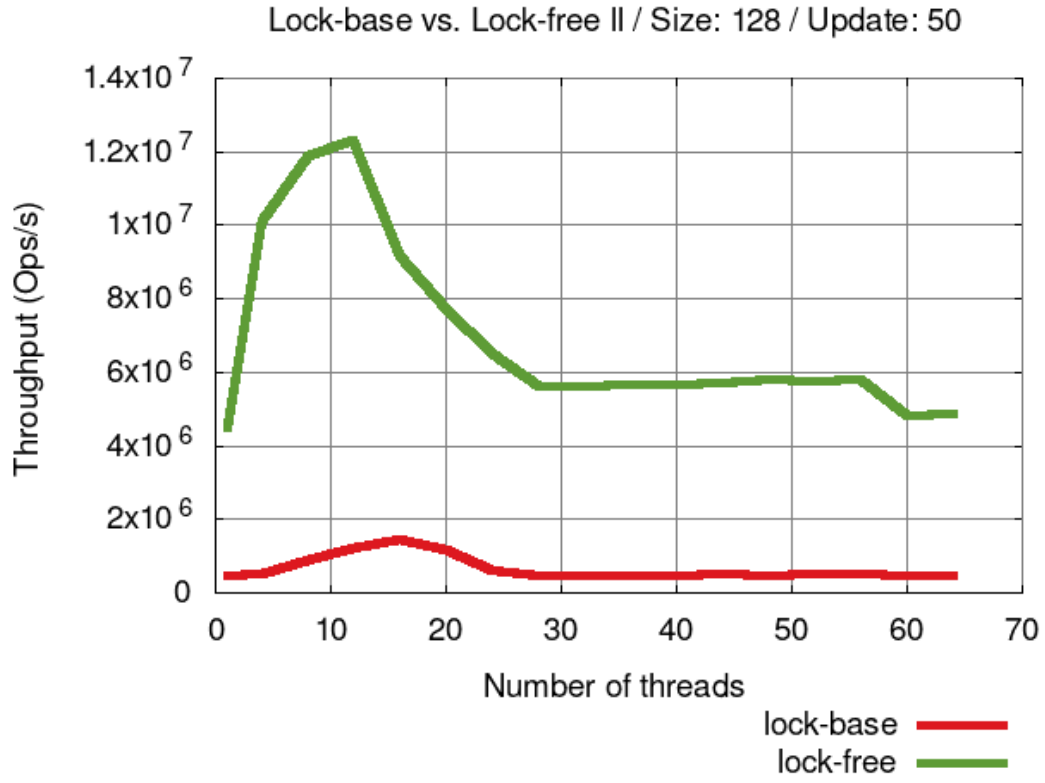

Lock-base vs. Lock-free II / Size: 128 / Update: 0
Throughput (Ops/s)
4.5x10 7
4x10 7
3.5x10 7
3x10 7
2.5x10 7
2x10 7
1.5x10 7
1x10 7
5x10 6
0
0 10 20 30 40 50 60 70
Number of threads
lock-base
lock-free
Lock-base vs. Lock-free II / Size: 128 / Update: 0
Scalability
8 7 6 5 4 3 2 1 0
0 10 20 30 40 50 60 70
Number of threads
lock-base
lock-free

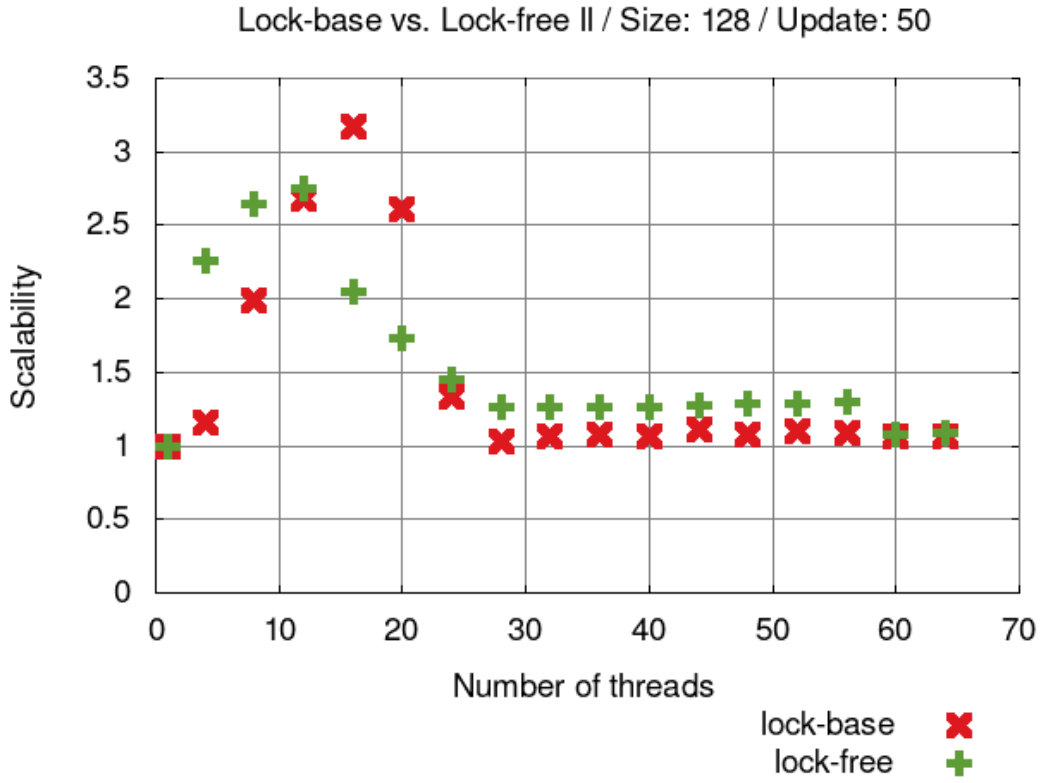
實驗結果


Lock-base vs. Lock-free II / Size: 128 / Update: 10
Throughput (Ops/s)
3.5x10 7
3x10 7
2.5x10 7
2x10 7
1.5x10 7
1x10 7
5x10 6
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free

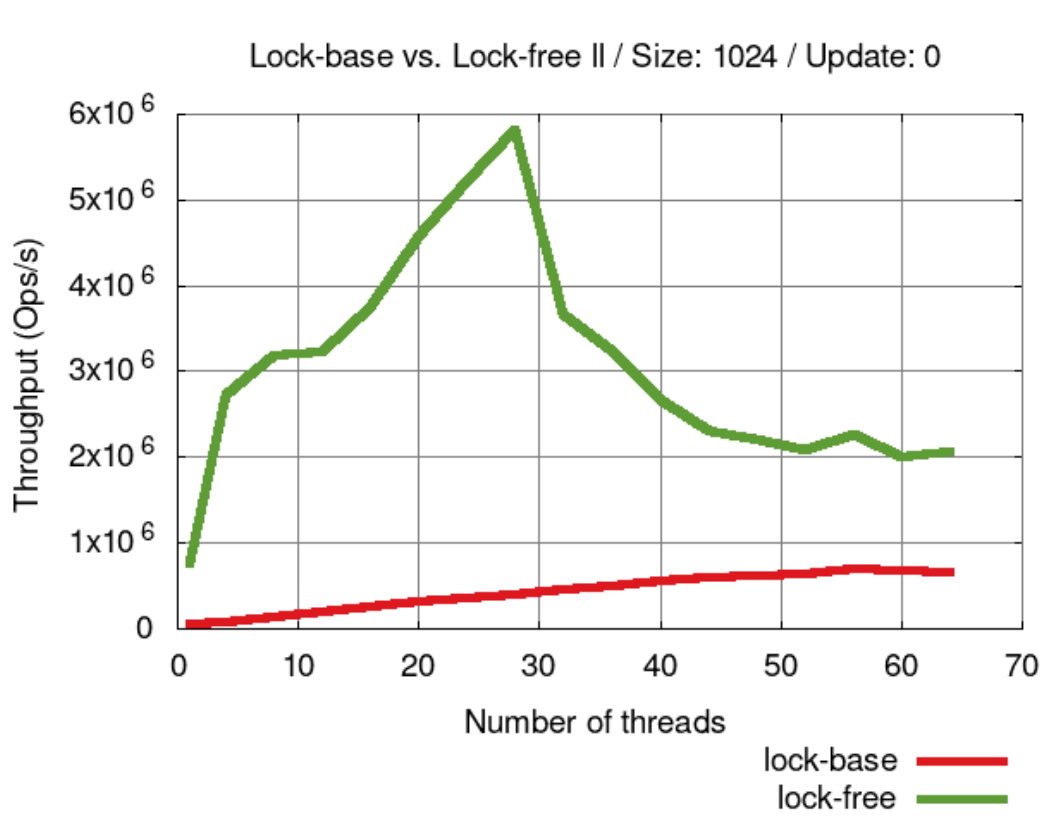

Lock-base vs. Lock-free II / Size: 128 / Update: 10
Scalability
7
6
5
4
3
2
1
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free

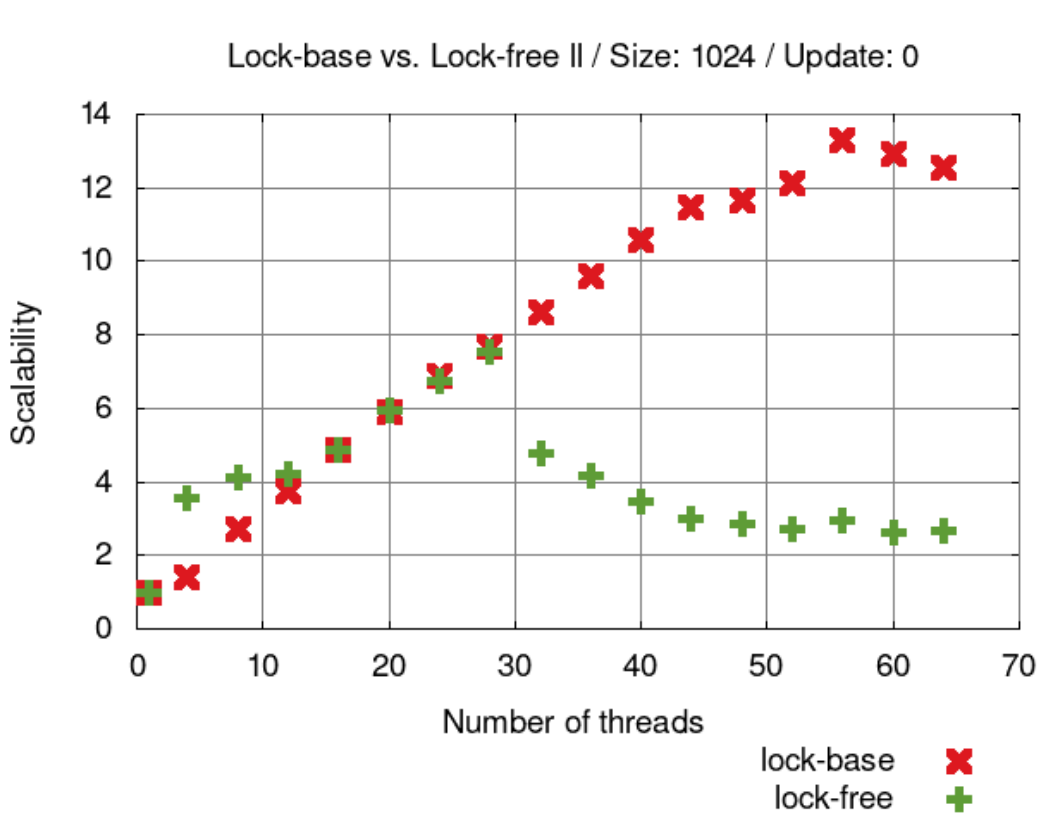
實驗結果


Lock-base vs. Lock-free II / Size: 128 / Update: 50
Throughput (Ops/s)
1.4x10 7
1.2x10 7
1x10 7
8x10 6
6x10 6
4x10 6
2x10 6
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free

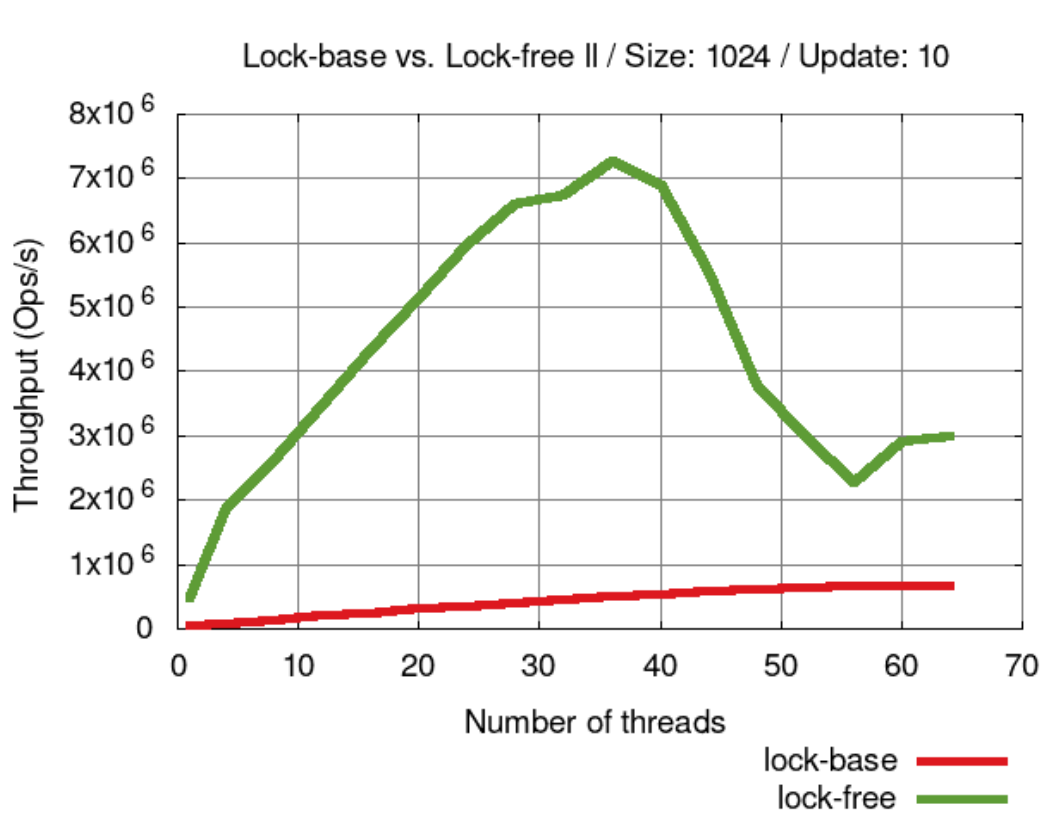

Lock-base vs. Lock-free II / Size: 128 / Update: 50
Scalability
3.5
3
2.5
2
1.5
1
0.5
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free

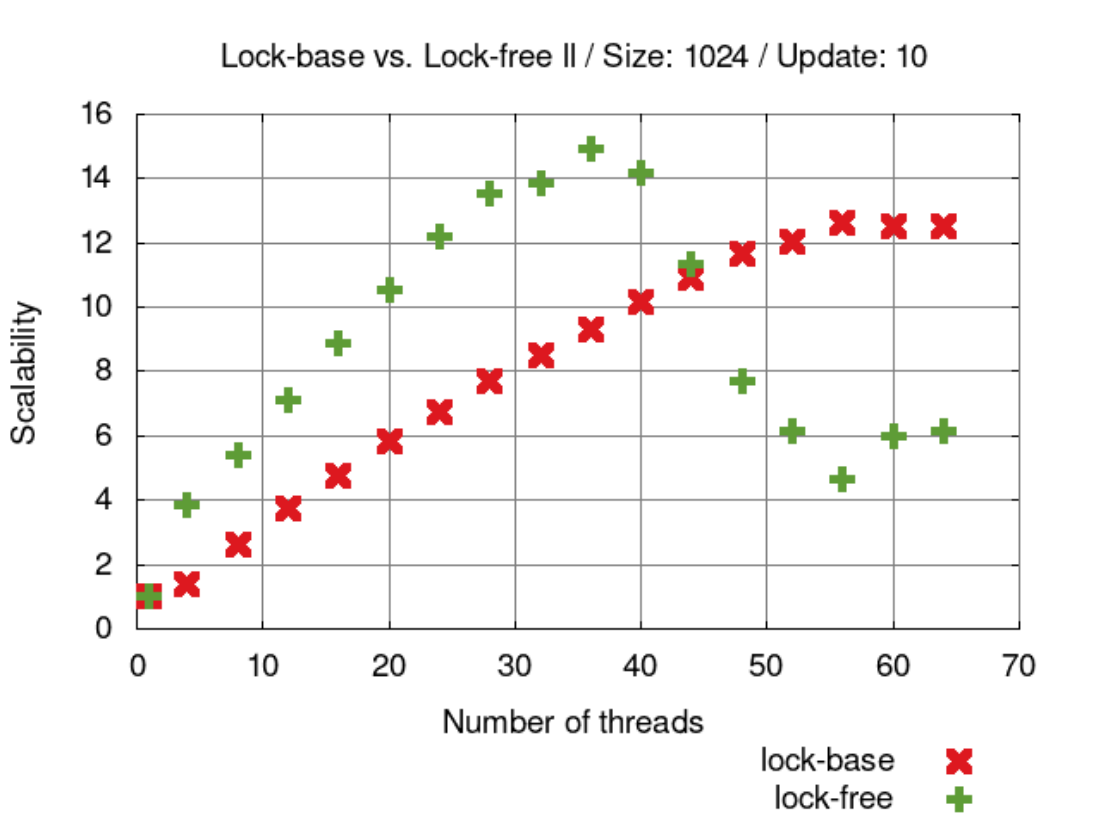
實驗結果


Lock-base vs. Lock-free II / Size: 1024 / Update: 0
Throughput (Ops/s)
6x10 6
5x10 6
4x10 6
3x10 6
2x10 6
1x10 6
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free

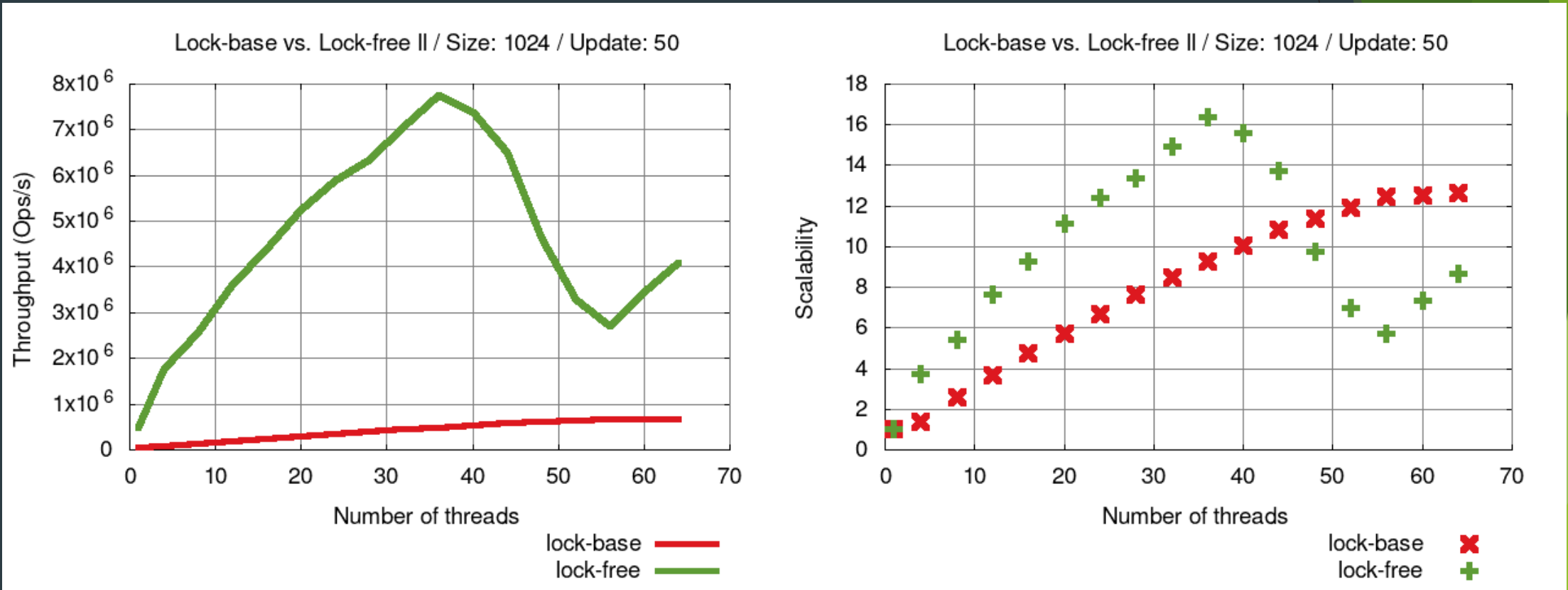

Lock-base vs. Lock-free II / Size: 1024 / Update: 0
Scalability
14
12
10
8
6
4
2
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free

實驗結果

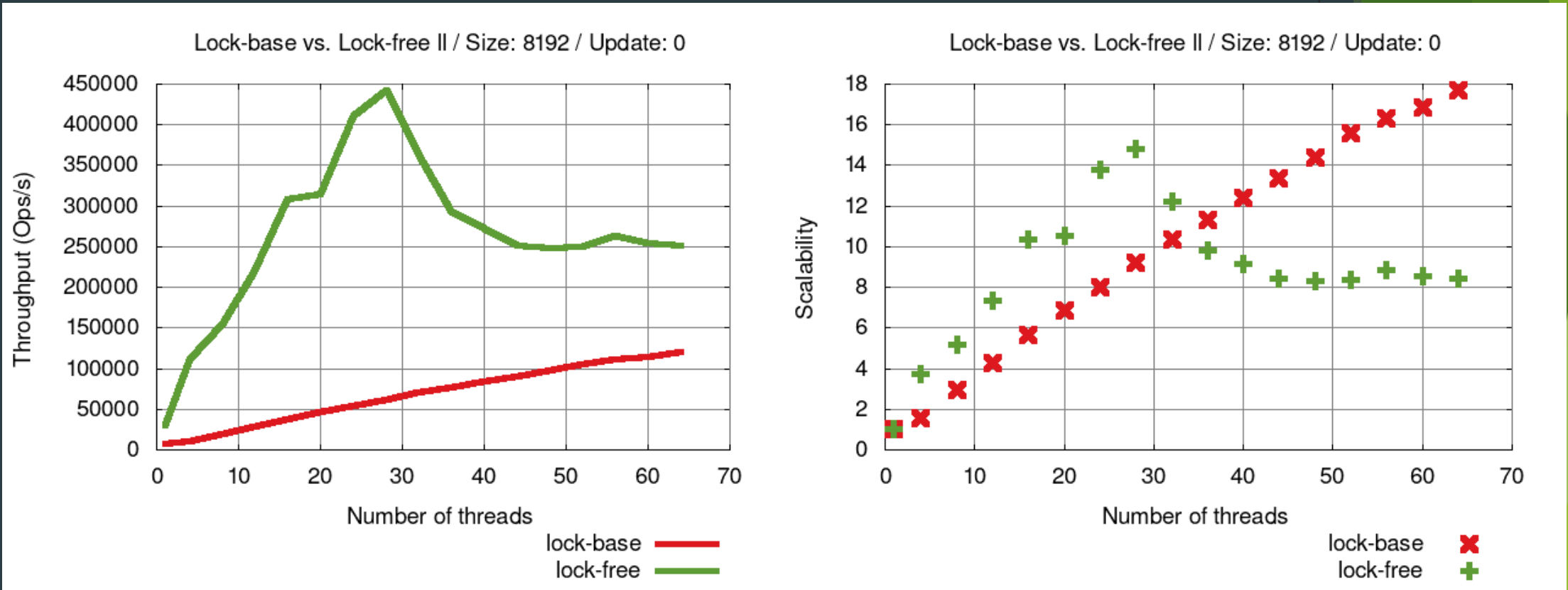

Lock-base vs. Lock-free II / Size: 1024 / Update: 10
Throughput (Ops/s)
8x10 6
7x10 6
6x10 6
5x10 6
4x10 6
3x10 6
2x10 6
1x10 6
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free


Lock-base vs. Lock-free II / Size: 1024 / Update: 10
Scalability
16
14
12
10
8
6
4
2
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free

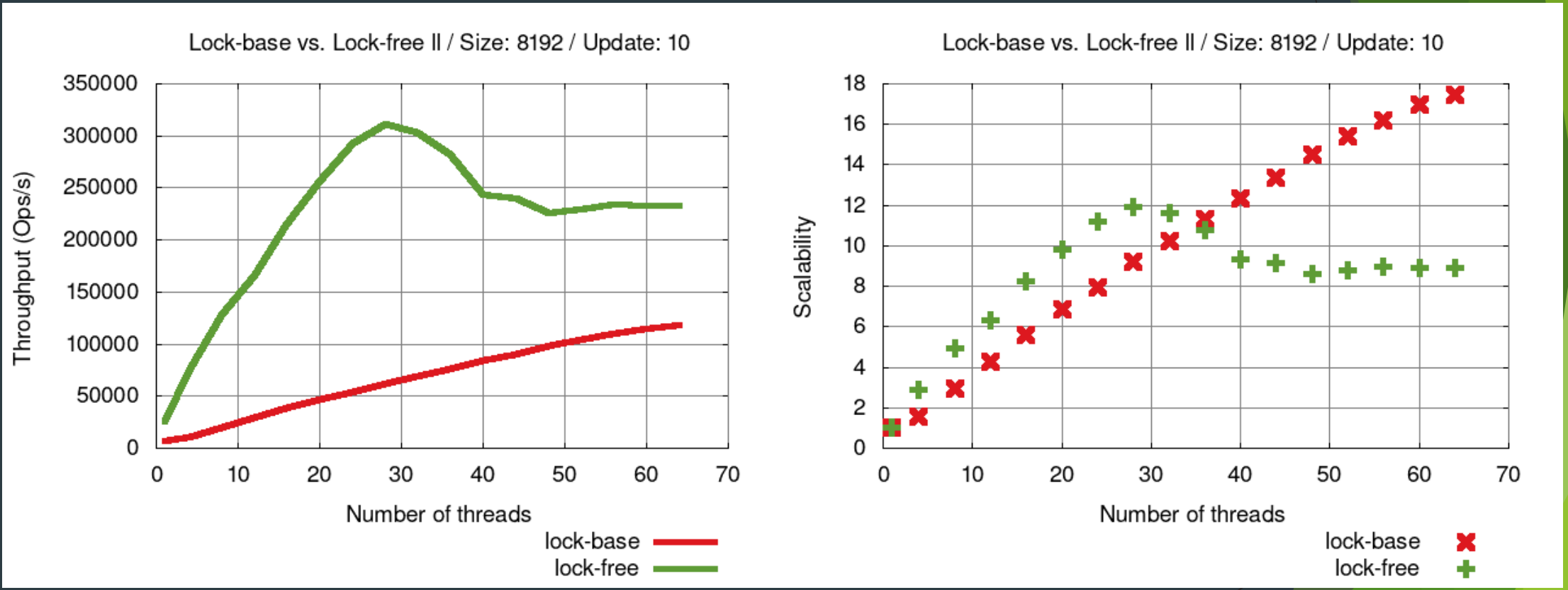
實驗結果


Lock-base vs. Lock-free II / Size: 1024 / Update: 50
Throughput (Ops/s)
8x10 6
7x10 6
6x10 6
5x10 6
4x10 6
3x10 6
2x10 6
1x10 6
0
0 10 20 30 40 50 60 70
Number of threads
lock-base
lock-free
Lock-base vs. Lock-free II / Size: 1024 / Update: 50
Scalability
18 16 14 12 10 8 6 4 2 0
0 10 20 30 40 50 60 70
Number of threads
lock-base
lock-free

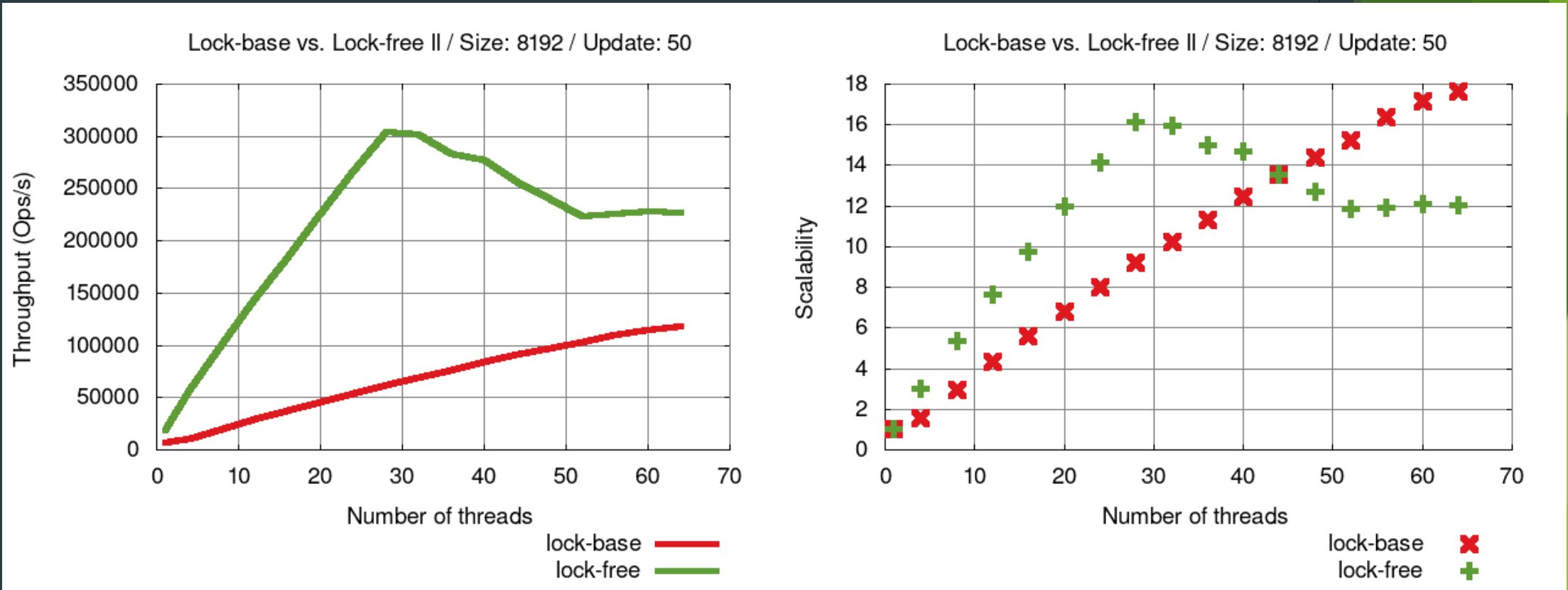
實驗結果


Lock-base vs. Lock-free II / Size: 8192 / Update: 0
Throughput (Ops/s)
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free
Lock-base vs. Lock-free II / Size: 8192 / Update: 0
Scalability
18
16
14
12
10
8
6
4
2
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free

實驗結果


Lock-base vs. Lock-free II / Size: 8192 / Update: 10
Throughput (Ops/s)
350000
300000
250000
200000
150000
100000
50000
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free
Lock-base vs. Lock-free II / Size: 8192 / Update: 10
Scalability
18
16
14
12
10
8
6
4
2
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free

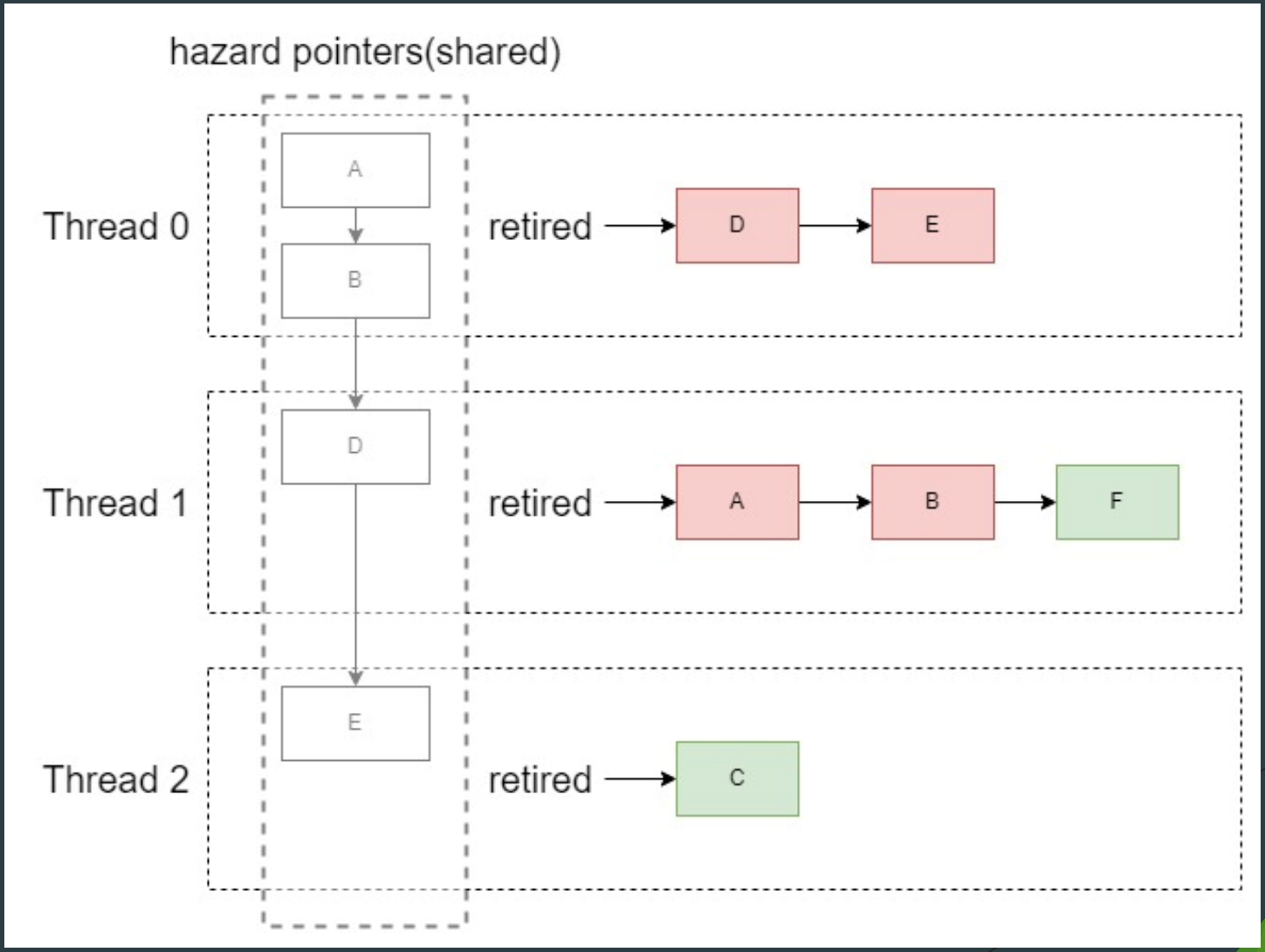
實驗結果


Lock-base vs. Lock-free II / Size: 8192 / Update: 50
Throughput (Ops/s)
350000
300000
250000
200000
150000
100000
50000
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free
Lock-base vs. Lock-free II / Size: 8192 / Update: 50
Scalability
18
16
14
12
10
8
6
4
2
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free

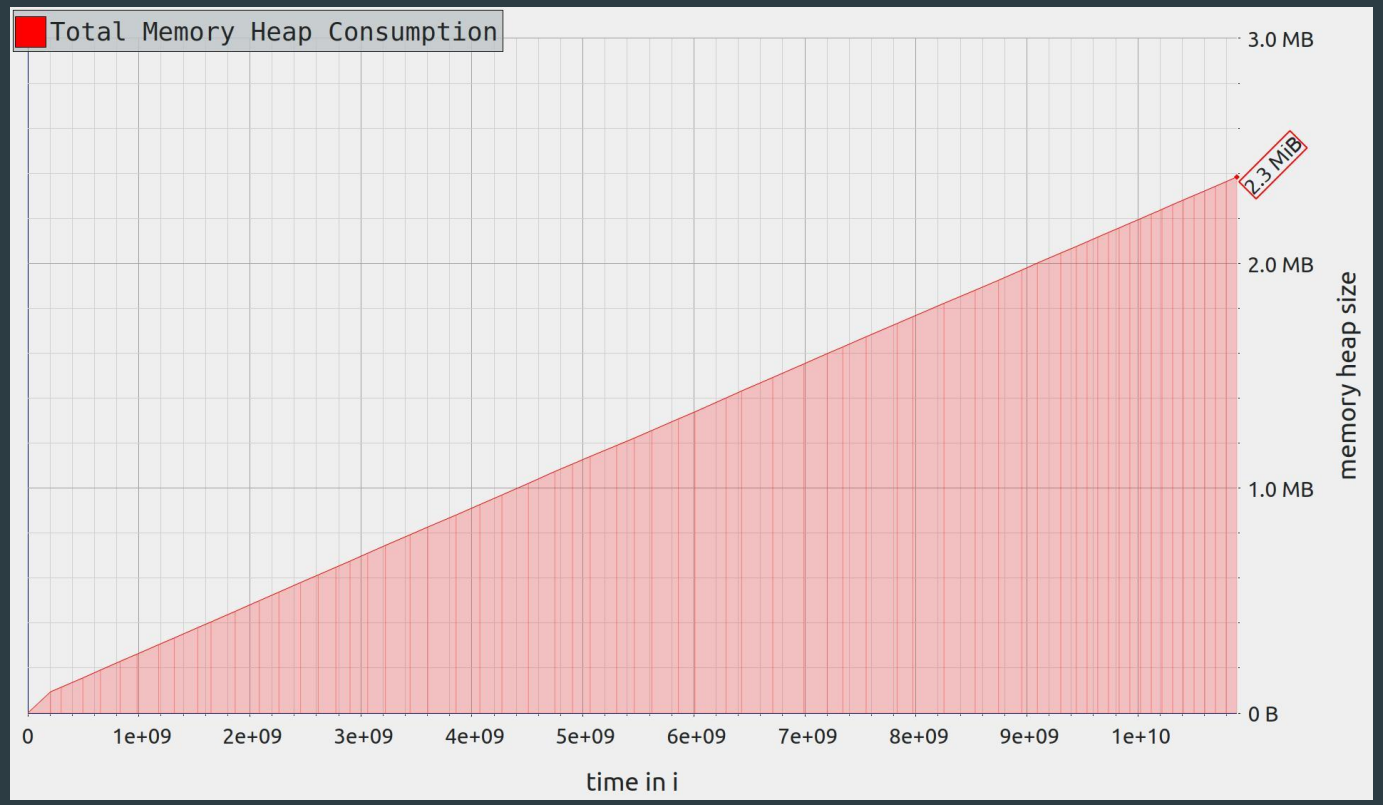
研究內容 Memory reclamation

- Lock-free 程式在回收動態記憶體時較 Lock-base 困難
- 需確保待釋放的記憶體沒有其他執行緒在讀取
- 實作論文 Lock-Free Data Structures with Hazard Pointers 並引入

Hazard pointers



實驗結果:未使用 hazard pointers

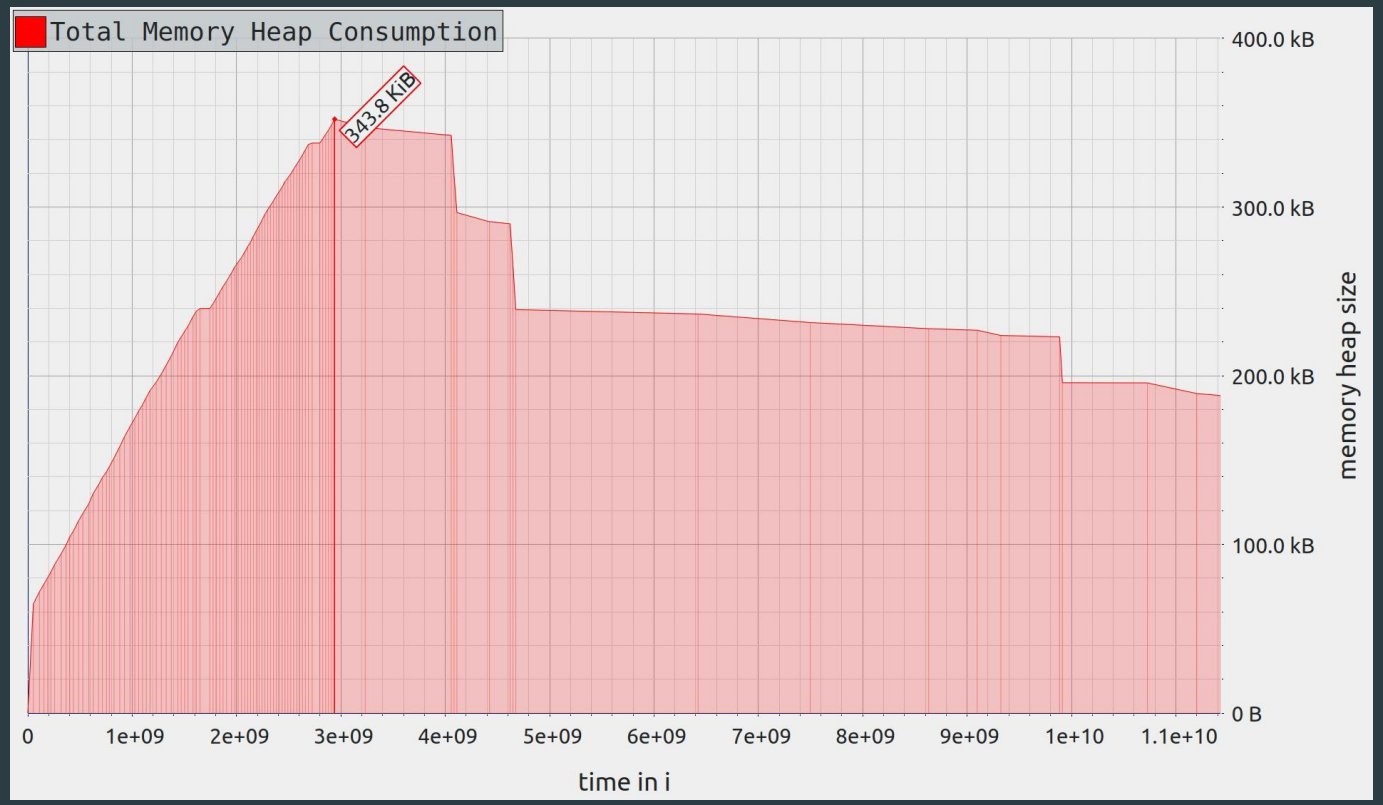


Heap size 呈線性增長

最大超過 2 MB

工具：valgrind

實驗結果:使用 hazard pointers



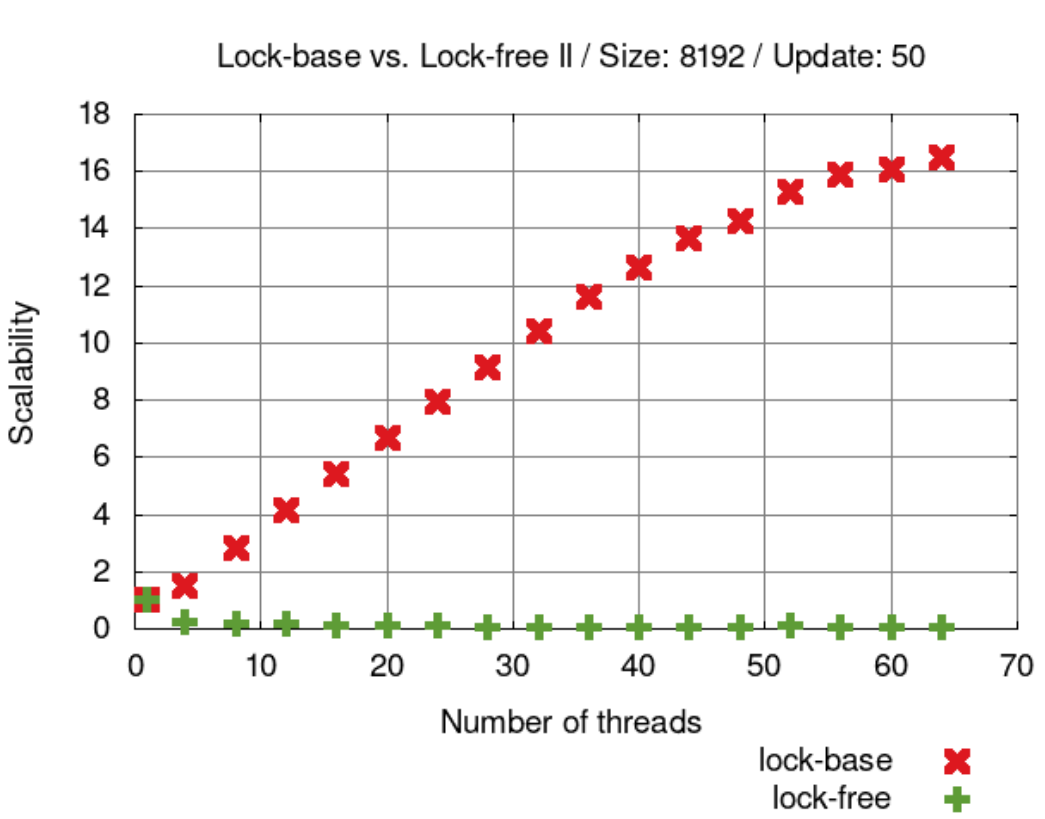
Heap size 受到限制

不超過 400 KB

工具：valgrind

實驗結果


Lock-base vs. Lock-free II / Size: 8192 / Update: 50
Throughput (Ops/s)
160000
140000
120000
100000
80000
60000
40000
20000
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free


Lock-base vs. Lock-free II / Size: 8192 / Update: 50
Scalability
18
16
14
12
10
8
6
4
2
0
0
10
20
30
40
50
60
70
Number of threads
lock-base
lock-free

總結

- 在不考慮 memory reclamation 的情況下， lock-free 在 throughput 上會比 lock-base 表現還好

- 目前的 lock-free linked list 搭配 hazard pointers 實作效能極差，可能須考慮其他手段，或改善現有實作

Reference

- Lock-Free Linked Lists and Skip Lists
- Lock-Free Data Structures with Hazard Pointers
- https://github.com/sysprog21/concurrent-ll
- https://github.com/kdnvt/hazard-pointer
- https://github.com/kdnvt/concurrent-ll/tree/project

感謝聆聽