ISCAS 中国科学院软件研究所学术年会’2021
暨计算机科学国家重点实验室开放周

学术论文

死锁的高效检测

Low-Overhead Deadlock Prediction （ICSE 2020）

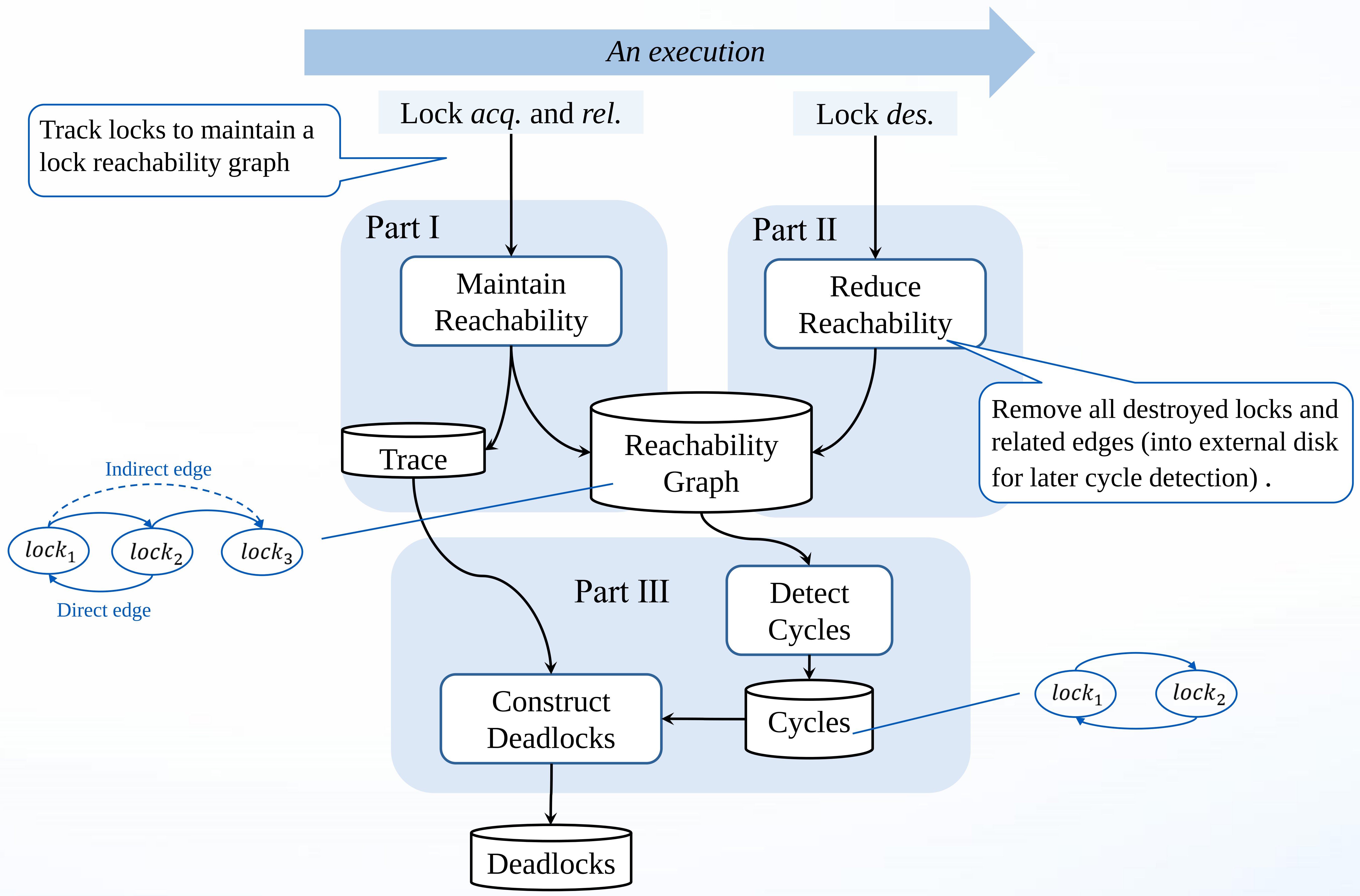
蔡彦（yancai@ios.ac.cn）、孟瑞杰 等 • 中国科学院软件研究所，计算机科学国家重点实验室

实时死锁
预测/检测？

死锁是普遍存在的一种并发缺陷，现有的具代表性的死锁检测工具MagicLock和UnDead不适用于即时检测，因为它们引入了较大的时间开销（100-1000×），这对终端用户来说是无法接受的，**如何高效地检测死锁**是并发程序分析中的一个难题。

我们提出了一种针对死锁的**即时高效**检测方案**AirLock**，大大提高了死锁的检测效率。实验表明，**AirLock的平均时间开销为3.5%**，该方案既可应用于开发内部测试，也可应用于已部署软件。

An overview of AirLock



Track locks to maintain a lock reachability graph

Remove all destroyed locks and related edges (into external disk for later cycle detection) .

已有工作 直接驱动一个指数复杂度算法去分析程序运行的trace，效率低。

AirLock AirLock 将死锁检测分为两部分。首先，通过实时建立任何两个锁上的可达关系图，并运行一个多项式复杂度的算法，将所有相互可达的锁找出来；若没有此类锁，则不存在死锁。否在，AirLock将可能涉及到死锁的这些锁找出来，并将它们涉及的同步事件找出，之后使用一个指数复杂度的算法来判断具体的死锁。这一设计，不仅仅降低了死锁检测的耗时，更使得这种方案可以和软件一起打包，部署在终端，在软件部署会持续做死锁检测。

Experiment

□ 有效性和效率

Technique	#Deadlocks	Time overhead (average)	Memory consumption (average)
UnDead	6	37121.1%	250 MegaBytes
MagicLock	6	3193.0%	130 MegaBytes
AirLock	6	**3.5%**	100 MegaBytes

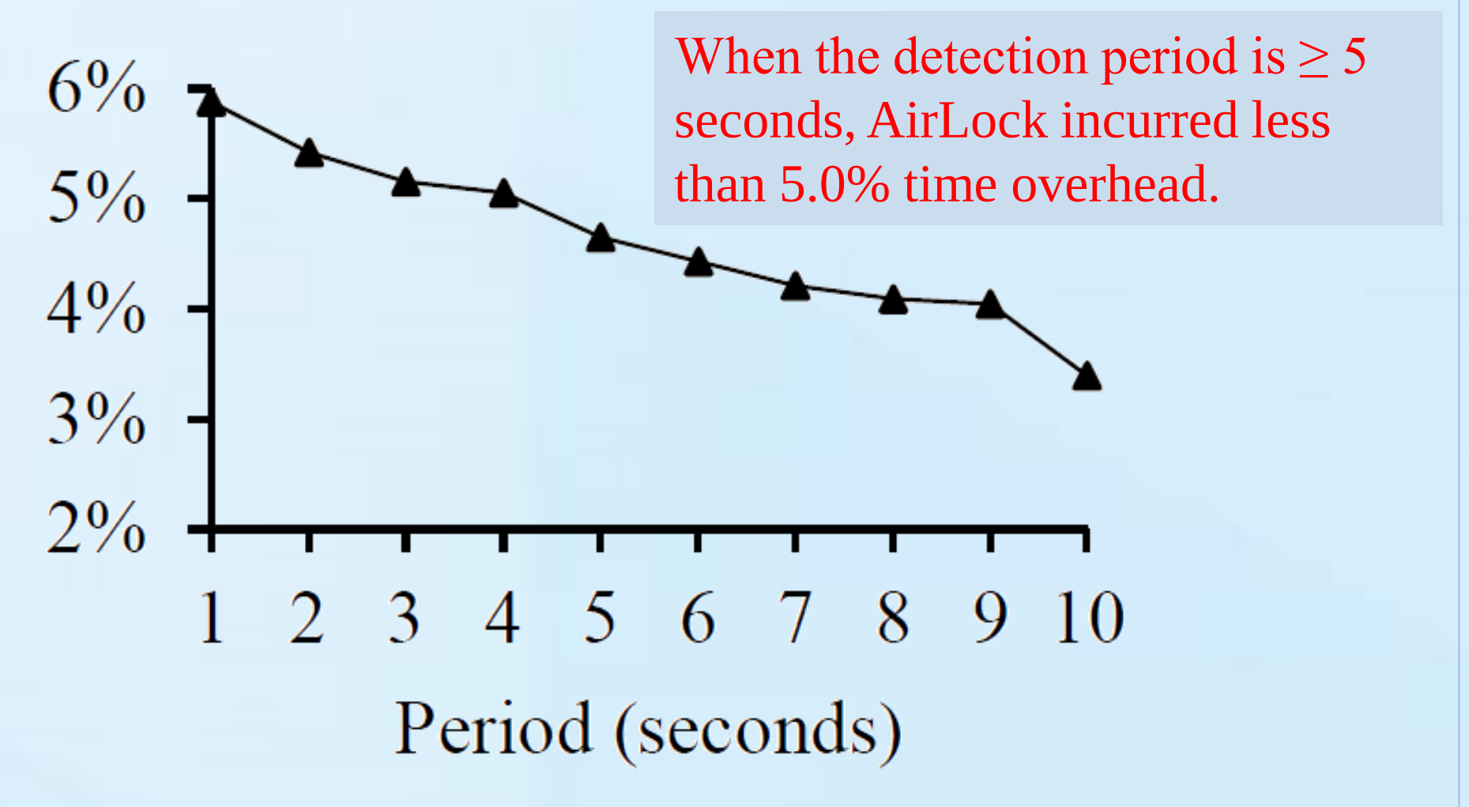
The three techniques detected the same number of deadlocks.

Low time overhead

Low memory overhead

□ 高频死锁检测下的耗时对比

	Native	UNDEAD	MAGICLOCK	AIRLOCK
MySQL1	36.8s	>10 Hours	39.0s (106.1%)	1.1s (3.1%)
Firefox	100.4s	>10 Hours	324.7s (323.4%)	3.0s (3.1%)



When the detection period is ≥ 5 seconds, AirLock incurred less than 5.0% time overhead.

Overhead of AirLock on Firefox with periodical detections

When both benchmarks run for about 36.8 or 100.4s, both UnDead and MagicLock incurred larger overhead than AirLock.

Such detection period (>5 seconds) is already highly frequent for long-running programs.

Yan Cai, Ruijie Meng, and Jens Palsberg. 2020.
Low-Overhead Deadlock Prediction.
In Proc. ICSE 20, May 23–29, 2020, Seoul, Republic of Korea.
ACM, New York, NY, USA,12 pages.
https://doi.org/10.1145/3377811.3380367

蔡彦
中国科学院软件研究所
计算机科学国家重点实验室A216
http://yancai.site
yancai@ios.ac.cn
010-62661613