

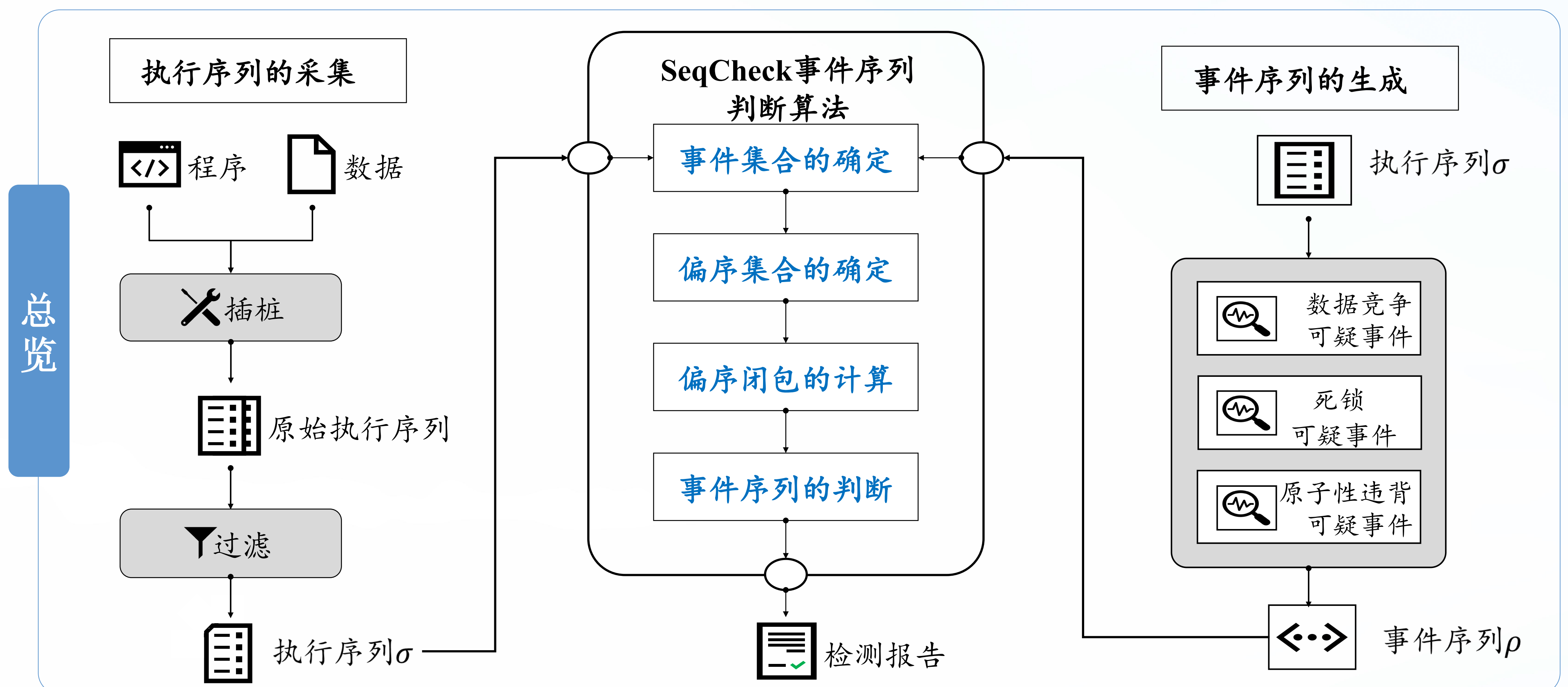
## 精确高效的并发缺陷统一动态检测技术

## Sound and Efficient Concurrency Bug Prediction

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

并发缺陷是并发程序中最常见的问题, 引发的事故屡见不鲜, 其危害不容小觑。现有的并发缺陷检测技术或存在误报问题, 或执行速度缓慢, 很难在检测能力和检测效率上两全。并且, 许多技术仅针对特定的缺陷类别设计, 缺乏通用性。

针对现有技术的不足, 本文提出一种**高效且无误报**的动态检测方法**SeqCheck**, 能够**通用**于多种并发缺陷。在数据竞争的检测实验中, SeqCheck相比最新先进技术多检测出**16%**的缺陷数量, 平均执行速度是其他技术的**12**倍。



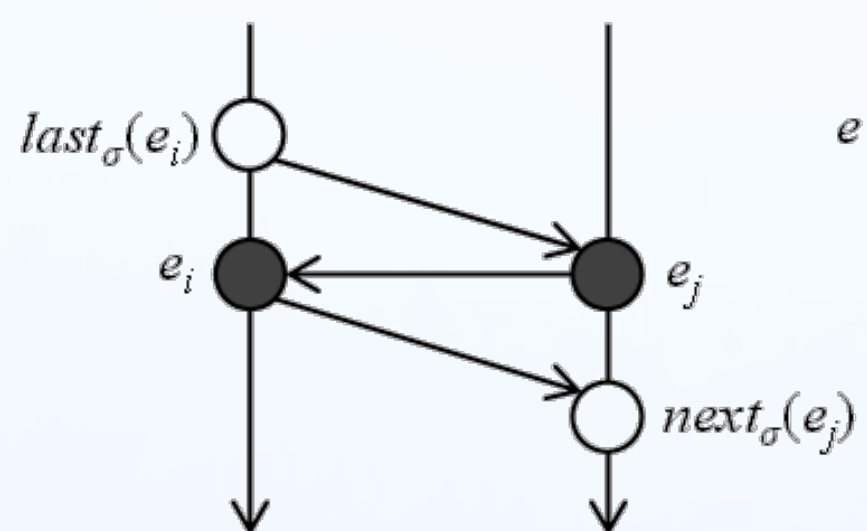
SeqCheck事件序列判断算法是SeqCheck的主算法, 具有 $O(n^2 \times \log(n))$ 的时间复杂度与 $O(n)$ 的空间复杂度。其接收执行序列 $\sigma$ 和事件序列 $\rho$ , 若 $\rho$ 能够真实发生, 则返回一个能够证实 $\rho$ 发生(包含子序列 $\rho$ )的合法执行序列 $\sigma_1$ 。该算法首先确定 $\rho$ 发生所必须的时间集合, 然后由四种偏序规则构造事件必须满足的偏序集合, 继而对偏序集合计算偏序闭包, 最终对 $\rho$ 的合法性进行判断。若在闭包计算过程中无矛盾产生, 则给出肯定判断。

## 算法示意

## 常见并发缺陷检测

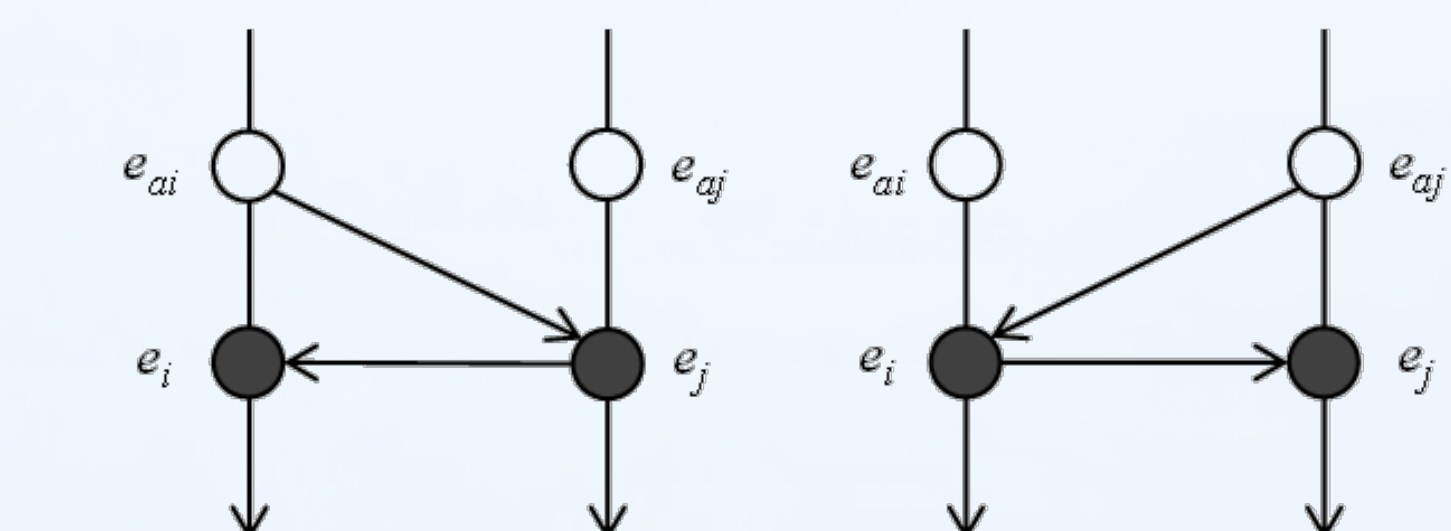
## ➤ 数据竞争

$$\mathcal{A} = \{(e_i, e_j)\}$$



## ➤ 原子性违背 (略)

## ➤ 死锁: 二者中至少一个可发生



## 算法 7: DetectBug

```
1 Function DetectDataRace( $\sigma$ ):
2   foreach potential bug  $(e_i, e_j) \mid e_i, e_j \in \mathcal{E}_\sigma^r \cup \mathcal{E}_\sigma^w, e_i \propto e_j$  do
3      $\rho \leftarrow \langle last_\sigma(e_i), e_j, e_i, next_\sigma(e_j) \rangle$ 
4      $\mathcal{A} \leftarrow \{(e_i, e_j)\}$ 
5     if SeqCheck( $\sigma, \rho, \mathcal{A}$ ) then
6       print A data race detected.
7 Function DetectDeadlock( $\sigma$ ):
8   foreach potential bug  $(e_i, e_j) \mid e_i, e_j \in \mathcal{E}_\sigma^{req}$  do
9      $e_{ai} \leftarrow$  the event such  $e_{ai} \in ls_\sigma(e_i), oid(e_{ai}) = oid(e_j)$ 
10     $e_{aj} \leftarrow$  the event such  $e_{aj} \in ls_\sigma(e_j), oid(e_{aj}) = oid(e_i)$ 
11     $\rho_1 \leftarrow \langle e_{ai}, e_j, e_i \rangle$ 
12     $\rho_2 \leftarrow \langle e_{aj}, e_i, e_j \rangle$ 
13    if SeqCheck( $\sigma, \rho_1, \emptyset$ )  $\vee$  SeqCheck( $\sigma, \rho_2, \emptyset$ ) then
14      print A deadlock detected.
15 Function DetectAtomViol( $\sigma$ ):
16   foreach potential bug  $(e'_w, e_w, e_r) \mid e'_w, e_w \in \mathcal{E}_\sigma^w, e_r \in \mathcal{E}_\sigma^r$  do
17      $\rho \leftarrow \langle e'_w, e'_w, e_r \rangle$ 
18      $\mathcal{A} \leftarrow \{\dots\}$ 
19     if SeqCheck( $\sigma, \rho, \mathcal{A}$ ) then
20       print An atomicity violation detected.
```

数据竞争  
检测算法死锁  
检测算法原子性违反  
检测算法

## 实验验证

## 有效性 (数量)

## 效率 (时间)

数据竞争	SeqC	333	SeqC	0.5
	SHB	144	SHB	5.95
	WCP	130	WCP	6.6
	M2	269	M2	11.6
	SyncP	285	SyncP	6.65
死锁	SeqC	20	SeqC	0.00 (5.95 s)
	Dirk	21	Dirk	1.03
原子性违背	SeqC	30	SeqC	0.32
	AtomFuzzer	0	AtomFuzzer	0.45

SeqCheck检测到数据竞争的数量**最多**, 且覆盖到了其他工具的所有检测结果, 检测效率上也有前所未有的**显著提升**; 在死锁实验和原子性违背实验中, SeqCheck同样表现出良好的检测能力和检测效率。实验结果证实了SeqCheck优于现有先进技术。