Graph Theory and Program Analysis

Graph-theoretic methods in database theory

Mihalis Yannakakis

Secure programming via game-based synthesis

William Harris

Today's Topic: 2013 ACM Turing Award Goes to Leslie Lamport for Advancing Reliability and Consistency of Computing...

a document preparation system that is the de facto standard for technical publishing in computer science and other fields.

Matrix Multiply on Semiring

- A closed semiring
  - 0 and 1 exists
  - product x is associative, closed under finite products
  - sum + is associative/commutative, closed under finite sums
  - x distributes over +
- A graph
  - 0 is no path, 1 is null path
  - sum combines multiple paths
  - product extends a path

\[
\text{for } k := 1 \text{ to } n \text{ do for all } i, j \text{ do}
\]
\[
A_{ij}^{0} := A_{ij}^{0} + A_{ij}^{0} \times (A_{ij}^{0})^{*} \times A_{ij}^{1}
\]

Reachability

- Problem
  - \(A_{ij} = 1\) iff there is a path from i to j
- Solution
  - sum is \(||\), product is \&
  - initial \(A_{ij} = 1\) if i,j connected; otherwise 0
- Explanation
  - connectivity including node k is the connectivity with node 1 to k-1, plus the connectivity from i to k, and plus the connectivity from k to j
  - Transitive closure can then be computed easily

\[
\text{for } k := 1 \text{ to } n \text{ do for all } i, j \text{ do}
\]
\[
A_{ij}^{k} := A_{ij}^{k-1} + A_{ij}^{k-1} \times (A_{ij}^{k-1})^{*} \times A_{ij}^{1}
\]

Database Queries

- Relational databases
  - "directed" hypergraphs, with labelled edges
  - a path "spells" a word
  - an L-path if the spelled word is in the language L
- Recursive queries
  - need to compute the transitive closure
  - not expressible in relational algebra/calculus
  - extensions in datalog and graph-oriented query languages
- Datalog language
  - example \(p(X,Y) := q_{1}(X,Z_{1}), q_{1}(Z_{1},Z_{2}), ..., q_{1}(Z_{n},Y))\).
  - variables, predicates, recursion

L-path

- A specification L
  - a regular expression
    - e.g. lock acquire/release/error DFA in SLAM
    - e.g. even-length paths
  - a context-free grammar
    - e.g. legal interprocedural paths in optimization
  - A path satisfies L is an L-path
  - SLAM
    - is there an L-path that reaches the error state anywhere in code?
    - is it an MOP problem?
- Interprocedural analysis
  - what is MOP invariance at every point?
# Lock Safety

```c
do {
    KeAcquireSpinLock();
    if (*)
        do {
            KeAcquireSpinLock();
        }
    else
        do {
            KeReleaseSpinLock();
        }
} while (*);
KeReleaseSpinLock();
```

RE specification?

---

## Type State Analysis (one slide!)

Languages. That is, we construct a graph $H$ whose nodes are pairs $s\in S$ of $s\in S$ and $a\in A$, and which has an arc labelled $a$ from a node $(s,a)$ to another node $(s',a')$ if $M$ has a transition on letter $a$ from state $s$ to state $s'$ and $G$ has an arc from $a$ to $a'$. (Actually, the labels in the product graph are not important.) Let $s_0$ be the initial state of $M$ and $F$ its set of accepting states. Then, the database graph $G$ contains an $L$-path from a node $s$ to a node $y$ iff $(s_0,y)$ can reach in the product graph a state $(s,Y)$ with $s_0, a$. For $F$.

- A state space $H$ for lock-safety analysis
  - $(s,a) \rightarrow (t,b)$ if $s\rightarrow t$ and $a\rightarrow b$
  - problem: if (start, unlocked) can reach any $(s, error)$ in $H$
  - single-source transitive closure

---

## Type interpretation

### Definition (Type interpretation)

The type interpretation $T[E]$ compositionally maps a regular expression $E$ to the corresponding simple type:

- $T[\epsilon] = \emptyset$ (empty type)
- $T[\{\}] = \{\}$ (unit type)
- $T[\{a\}] = \{a\}$ (singleton type)
- $T[E \times F] = T[E] \times T[F]$ (product type)
- $T[U] = \{\{v_1, \ldots, v_n\} \mid v_i \in T[E]\}$ (list type)

---

## Weaver PL results

- Automatically reproduced manually-written programs and automatically wrote new ones from small, declarative policies that forbid known vulnerabilities
- Reduced weaving to game-based synthesis, and proposed and evaluated a novel symbolic game-solving algorithm [CAV ‘12]

---

## Program Analysis as Queries

- **Queries**
  - proposition logic
  - regular
  - recursive
    - e.g. lock safety spec
    - context-free
    - e.g. datalog
  - Type of queries
    - existential: does something exist?
    - universal: does the property always hold?

---

## Secure programming via game-based synthesis

[Diagram of Lock Safety]

William Harris

[Diagram of Type State Analysis]

[Diagram of Type interpretation]

[Diagram of Weaver PL results]

URCS Department Seminar, March 2014
Today's Topic: 2013 ACM Turing Award Goes to Leslie Lamport for Advancing Reliability and Consistency of Computing Systems

Tuesday, March 18, 2014

ACM has named Leslie Lamport, a Principal Researcher at Microsoft Research Silicon Valley, the recipient of the 2013 ACM A.M. Turing Award for improving clear, well-defined coherence on the seemingly chaotic behavior of distributed computing systems, in which several autonomous computers communicate with each other by passing messages. He devised important algorithms and developed formal modeling and verification protocols that improve the quality of real distributed systems. These contributions have resulted in improved correctness, performance, and reliability of computer systems.

Lamport's practical and widely used algorithms and tools have applications in security, cloud computing, embedded systems and database systems as well as mission-critical computer systems that rely on secure information sharing and interoperability to prevent failure. By notions of safety, where nothing bad happens, and liveness, where something good happens, contribute to the reliability and robustness of software and hardware engineering design. His solutions for Byzantine Fault Tolerance contribute to failure prevention in a system component that behaves erroneously when interacting with other components. His creation of temporal logic language (TLA+) helps to write precise, sound specifications. He also developed LaTeX, a document preparation system that is the de facto standard for technical publishing in computer science and other fields.

Program Security

- Security = Safety + Liveness
- OS Weaver
  - safety
    - i.e. ‘compress’ cannot have certain privileges
  - liveness
    - i.e. ‘compress’ can still be used correctly
  - Xiaochen’s question at the talk
- Reachability analysis
  - can check for safety
  - can it insert code as OS Weaver can?
  - what about liveness?