SYNASC 2019 (& FROM 2019)

Notes by J.H.Davenport

4–7 September 2019

Chapter 1

4 September 2019

1.1 Welcome

DZ Brief start.

DP Welcome to the University, overview. Also SYNASC, since 1999. Chairs from many countries since 2009.

Hong 78 submissions; 47 for presentation, of which 19 directly for publication, and 28 revised.

1.2 On the nature of Symbolic Execution: Bonsangue

At Leiden, we teach a course “what is testing” wth Frank de Boer. Claims 30–85 faults/1000lines of code. 66% of errors are not discovered until operational. Scale from manual testing to deductive verification: increasing confidence and complexity. Symbolic execution in the middle, just below static analysis. [King-CACM1976]. Why was this forgotten? It was expensive (better hardware, also better software, notably SMT). Generate test cases by counterexample queries.

Symbolic execution should use expressions from the base language. The semantics are complex in detail. Correctness: for each reachable symbolic configuration and every state which satisfies the path conditions, there exists a corresponding concrete execution.

Creating a new object requires a fresh variable.

The symbolic semantics of method call are technical. Various implementation details matter (branch immediately on heap allocation?). Backward symbolic execution is correct by default (but not very popular).

Libraries and native code are an issue: Concolic [concrete+symbolic] is needed.

Q Subtypes?

A Could be done.

1.3 Efficient Validation of FOL$_{ID}$ reasoning: Stratulat

Soundness checking of cyclic pre-proofs. Extend Gentzen's LK system. $\frac{\Gamma\vdash\Delta}{\Gamma[\delta]\vdash\Delta[\delta]}$ (Substitution rule).

A pre-rpoof is a finite derivation tree with backlinks (bud-companion relationships). using CLKID (LK+'=' rules + unfold + case). [BrotherstonSimpson2011]. Following them, we annotate proofs with traces. Essentially infinite descent, knwing that showing a true fact requires infinite unfolding is a contradiction. Hence need to check the inclusion reation between two Büchi automata. This is decidable, but doubly exponential.

A trace followin some (potentially infinite) path $p[N^1, N^2, \ldots]$ in a pre-proof tree is a sequence $(\tau_i)_{(i\geq 0)}$ of IAAs such that

1. τ_{i+1} is $\tau_i[\{x \mapsto u\}]$ if $S(n^i) \equiv (\Gamma, x \vdash \Delta?)$
2. Other rules

A preproof is a proof if every infinite path has an infinitely progressing trace starting from some point.

Hence a checking procedure

1. Normalis e P to a path-equivalent tree such that ...
2. Verify ...

yclist is the theorem-prover of [Brosterstonetal2012]. Integrates the Spot model-checker [Duret-Lutzetal2016]. Proofs are developed depth-first (many calls of Spot).

Our E-cyclist is an extension with our checker as an alternative. Complexity is polynomial. Using the benchmarks from [Brosterstonetal2012] get a speedup ranging from 1.43 to 5[1]. But our procedure is only a semi-decision procedure.

This is integrated into Coq. Use Noetherian induction over well-founded posets. Our cetificationmethodis based on spike [Stratulat2017b]. Use a well-founded ordering based on syntactic ordering of individual terms. But it's an open question whether we can always build an ordering.

1.4 Dramnesc/Jebelean

Go from specification to algorithms and specifications of sub-algorithms. Looking in particular at lists and multisets as examples. Hence "two lists have teh same elements" is easily expressible. $P[X, Y] : (I(X) \Rightarrow O(X, Y))$. Find a function F such that $\forall_X O(X, F(X))$. Use Theorema.

[1]But times were in ms: 1–400. Speaker was asked for bigger examples: not yet.

Assume elements $a, b, x, y, \dots$ which are otally ordered by $<$. Lists $U, V, \dots$. Inductive domain, and extend $<$. Multisets with $\mathcal{M}$. Sorting $\forall_X \exists_Y \text{IsSorted}(y) \wedge \mathcal{M}(X) = \mathcal{M}(y)$.

Various proof techniues: classical and novel. Infernece rules by unitpropagation, expand/compres multisets by eqality rewriting. Two constants generate a proof by cases $a \leq b | a > b$. Several strategies. Cover set, Noetherian induction, cascading (trigger synthesis of auxiliary algoithms) etc. Induction for binary functions is more complex. Cascading (conjectur generation): Skolem constants become universally quantified; metavariables from the goal become existentially quantified. Also "pair multisets" If the goal includes $\mathcal{M}[Y*] = \mathcal{M}[t_1] \cup \mathcal{M}[t_2] \cup \cdots$, transform the using into some $\mathcal{M}[t]$. For example if $M[V^*] = \{\{x\}\} \cup \mathcal{M}[X] \cup \cdots$ and x, X are incomparable, then we split $X = X_1 \cup X_2$ with $X_1 < x \leq X_2$.

Example 1 (Sorting) *We can apply "cover set" to*

1. *Skolemised variable, and this essentially synthesis MinSort, synthesising the sub-algorithms*

2. *On the input. On the non-null case, we can use either induction or split. Induction will basically give insertion sort. Split will basically give us Quicksort (with a_1 as pivot).*

There's also a way on synthesising mergesort.

1.5 Davenport

See `http://people.bath.ac.uk/masjhd/Slides/FROMSlides-post.pdf`.

1.6 Design and Validato of Coud Storage Systems using Rewriting Logic: Ölveczky

Note how dependent the world is on availability, e.g. of payment card systems, now we are close to cashless. For availability, we need data replication and partitioning.

Consistency Vital for financial information.

Eventual Consistency Good enough for Google, say. But not ebay.

Read-Atomic Multi-Partition Transactions (fractured read) — important for "friend" relationships (at least in the public eye)

SnapShot Isolation All see consistent data.

Causal Consistency If I see a reply, I must see the original.

So how do these get validated. We also need good performance. Testing + code reviews + informal proofs $\Rightarrow$ no guarantees. Hand proofs are error-prone and don't scale.

Hence formal methods. See [New14]. But which. We wanted rewriting logic. Then Maude for simulation and temporal logic. How to do performance analysis?

- Randomised simulations (OK, but ...)
- Statistical Model Checking (using PVeStA).

So for our megastore, we divide data into "entity groups" and work on these. Problem: can take hours.

Hence Cassandra, a facebook development in Timoșoara (?) . Still slower than Stonebraker's specialist tools, but correct by construction.

Note that AWS uses formal methods. Using Lamport's TLA+ and model checking. Praise for this for finding "corner cases" which get past code reviews etc. Even found a very subtle bug that would have led to lost data: simplest trace was 35 high-level steps. But this wasn't useful for performance degradation. Hence spekaer claims that Maude should be much better suited.

Hence claims that FM are an efficient way to design, test and describe industrial fault-tolerant distrbuted systems [at scale].

Q Open problems?

A Duplicate information. Component-based design.

1.7 (Skype)

Itroduction — two buyer protocol. So we are looking at Probabilistic Mltiparty Session Processes.. Sytax is is/hen/else, | (parallel) etc. $\sum_{i\in I} p_i s! \ldots$ sends values, also receives etc.

Theorem 1 $\Gamma \vdash P \triangleright \Delta$ *and* $P \equiv P'$ *implies* $\Gamma \vdash P' \triangleright \Delta$*;* $\Gamma \vdash P \triangleright \Delta$ *and* $P \rightarrow_{p_i} P'$ *implies ...*

1.8 Co-Inductive Proof

Context is Reachabiity language (Roșu et al.). These are language-parametric logic for programs. Specifies partial correctness, used for verification. There's a coinductive "floavour', but not really exploited. We wat to cast RL in a transition-system setting.

Smallest and greatest fixed point μF and νF. Induction is $X \subset F \rightarrow X \subset \mu F$ and Coinduction is converse. $\Rightarrow\diamond$ is "eventually reaches". Concept of a sub-system of a transition ssytem. Needs proper closure requirements.

"$\vdash$ is the greatest relatoin satisfying $\frac{T\vdash\partial I' \Rightarrow \diamond r}{\cdots}$" is the first proof system we have. See `https://hal.inria.rf/hal-01962912` for a use. This has completeness: $T \models \phi$ implies $T \vdash \phi$. Has logical compositionality, but $\vdash$ doesn't admit hypotheses.

Second system with $|\!\vdash$.

$$\frac{T|\!\vdash I \diamond m \qquad T, H|\!\vdash m \Rightarrow \diamond r}{T, h|\!\vdash r}$$

is the "big step rule" [Stp]. This has coinduction for [Stp] and induction for other rules.

$||\!\vdash$ defines our third system. Sound, complete and compositional.

Started in Coq, but coinduction is a pain. Hence Isabelle/HOL. This has Knaster–Tarski induction and co-induction.

1.9 Explaining SDN Failures via Axiomatisations: Georgiana Caltais

The aim is to explain failures in NetKAT [Smolka on NetKAT]. [Kozen1996] proposed the original KAT. Regular expressions with $. + *$ as operators. ($+ =$ disjunction, $. =$ conjunction).

```
switch=6.port=88.dest<-10.0.0.1 ..

pt=5.pt<-6 + pt=6.pt<-5
```

encodes that 5 and 6 are connected. Hence a network can be encoded in NetKAT, with interleaving switches (processing) and topology. Sound and complete axiomatisation [CJAndersonetal]. There are Kleene Algebra and Boolean Algebra axioms. Then can write network configuration programs in NetKAT. But how to we prove that these satisfy certain properties, e.g. no packet can go from 1 to 4. And if this isn't provable, why not?

Theorem 2 *If P is a policy on n switches, then $\vdash in.(p.t)^*.out$ is equvalent to $\vdash in.(p.t)^n.out$.*

Future work is looking at counterfactuals versus causality: what do I need to remove to solve the problem?

Q Implementation?

A Coming.

Q Dynamic topology?

A(supervisor?) not dynamic.

1.10 Bertini Tutorial: Hauenstein

See `http://www3.nd.edu/~jhauenst/SYNASC2019`.

Interested in solving equations over $\mathbf{C}$. N equations in n variables. Use *homotopy continuation*. I use Bertini, but there are others. See [Ste04]. Parallelised via MPI. Working on Bertini2 (C++/Python) but that's not quite stable yet. Main reason is overloading support.

If your system is over-determined, then use GB [BFS04] shows it's polynomial. Numerical algebraic geometry prefers well-constrained systems. "Solve" might be a formula (but runs out at degree 5). For $x^5 - x + 1$ Maple's `solve` gives five `\RootOf`, but `fsolve` gives only the real root.

```
variable_group x;
function f;
f=x^5-x+1
```

Note Bertini isn't actually a proof. But you get a lot of information back for verification. So for Bertini, "solve" is a numercail approximation plus a refinement algorithm. Can ask for higher precision than default.

How?

1.10.1 First consider a square isolated system.

1. Need to find a parameterised family: choose $a_5x^5 + \cdots + a_0$.
2. Need to create a homotopy. Start with $(2+3\sqrt{-1})(x^5-1)$. Then $H(x;t) = (1-t)f(x) + tg(x)$.
3. Follow the homotopy.

$x^2 + 2x - 8; xy + 2x + 4y - 3$ gives one real root and three paths that tend to infinity. So start with $x^2 - 1$ and $(x-2)(y-1)$, which we get by defining x and y to be in separate`variable_group` (or `hom_variable_group`). If there are natural groupings, we should certainly respect that. Finding the best grouping is NP-hard in theory.

Polyhedral (BKK) bounds [HS95]. Note that you can "make your own" homotopy if you want. This can extend to non-linear (in the parameters) homotopies.

[Seidenberg1954] says compute critical points of the Euclidean distance functions. His example had a condition number of 10^{22} so Bertini went to 96 digits.

1.10.2 All solutions?

Chapter 2

5 September 2019

2.1 Using Numerical Insights to Improve Symbolic Computations: Hauenstein

Interested in solving polynomial systems over the complexes (in general). Symbolic wants exact computations, hence expressions swell. Floating point trades this for rounding errors.

Example 2 *Cubic-centered 12-bar mechanism, with every vertex also linked to centre. Rigid bars and rotational joints. But 3D model lost in Frankfurt airport. Factor out the trivial motions by fixing origin, and vertice P_7, P_8. Variables are 18 coordinates (6 vertices), and we have 17 equations (sides have length 2, distances to centre $\sqrt{3}$). Hence there is freedom. Bertini: a 3-fold, 6 surfaces and two curves. In fact, only the curves describe useful motions (rest are degeneracies, apparently).*

Weil: generic points describe solution sets. Over $\mathbf{Q}$, $x^2 + y^2 = 1$ and $(\pi/4, \sqrt{1 - \pi^2/16})$ are equivalent. Example: $P_1 = P_4$, $P_2 = P_8$, $P_3 = P_5$ and $P_6 = P_8$. A complete Macaluay2 ran out of memory, but verifying this is trivial. There's another example that can move over $\mathbf{C}$, but is rigid over $\mathbf{R}$ (like $x^2 + y^2 = 0$).

Example 3 (Dynamical Systems) *Near a fixed point, we have either spiral or centre (periodic) behaviour. So which polynomial systems have periodicity (Hilbert 16th).*

Theorem 3 (Grobman(1959), Hartman(1960)) *Depends on real parts of eigenvalues. $\Re(\lambda) < 0$ is spiral in, > 0 is spiral out.*

Also, centre requires locally analytic first integral, which means a power series solution. Verifyingthis is infinitely many equations.Darboux implies we need only check up to K.

$$\dot{u} = -v + g; \dot{v} = u + g; \dot{w} = -w + g; g = a_1u^2 + a_2v^2 + a_3w^2 + a_4uv + a_5uw + a_6vw.$$

By leading terms, $H(u, v, w) = \cdots$. A complex conjugate pair of multiplicity 3 components appears at degree 4. Real points occur in $a_1 = a_2 = a_4 = 0$. Need to go to degree 7 for stabilisation. Therefore we show there are 7 componets (all linear) which can give a centre.

Example 4 (Matrix Multiplication) *Strassen: 7. What about approximate?*

$A_{2,1} = 0$ might require 6, but [Binietal] shows 5 to within error ϵ. [Landsberg2006] had gaps, see [Hauensteinetal2013a]. Really good slide explaining tensor model.

Approximation is about lying in the closure (either Euclodean or Zariski). Hence want to prove $M_2 \notin \overline{\sigma_6}$. Numerically, this was obvious, via Bertini. For a proof, need a polynomial that vanishes on $\overline{\sigma_6}$, but not M_2. Straight interpolation is impossible. Looked for a smaller space, and found them.

3×3 matrices open. Approximate is in [16,20]. [Smi13] shows 20 by least squares approximation. Exact is in [19,23]. [Lad76] shows 23, not improved on.

2.2

In a model M, each sort s is a set M_s, each symbol σ is a relation/function: $M_{s_1} \times \cdot M_{s_n} \to P(M_s)$ (could be multi-valued).

We extend $\rho : Var \to M$ to $\overline{\rho} : Pattern(\Sigma) \to ?$. Therefore $M \models \phi$ etc. A Specification (S, Σ, F) and ask for a (S, Σ, F)-model M such that $M \models \phi \forall \phi \in F$.

How does this extend to patterns? Can we interpret them in a traditional tw-valued way? Require that, for each $(s_1, s_2) \in S \times S$, Σ incluedes a distinguised symbol $\lceil - \rceil_{s_1}^{s_2}$ with axiom ....

Fairy obvious axiomatisation of product sorts. But I want to extend ML to talk about induction and coinduction. Hence Knaster–Tarski Theorem: least and greatest fixed points μF and νF. Hence "Matching μ-logic".

But in Applicative Matching Logic, we don't have sorts, or rather we have a universal sort. Proof of `rev(rev(l))=l`.

Mutually recursive types: how do we define Even and Odd?

I haven't spoken about the various proof systems for Matching Logics, encodings of other logics etc.

2.3 Verifying DPLL in Dafny: Ciobaca

SAT based on CDCL, which is based on DPLL. Example $(x_1 \vee x_2 \vee x_3) \wedge (x_3 \vee \neg x_2) \wedge (x_1 \vee \neg x_3)$. So if x_1 is false, x_3 must be false, and then x_2 must be true — *unit propagation*. CDCL adds clause learning (but maybe we learn too much). Then we add restarts etc. Even MiniSAT is > 5000 lines of C++. Bugs in SAT solvers [BLB10], and now requirements for UNSAT cores.

Used Dafny (OO language from Microsoft), a memory-safe language with pre/post-conditions checked by compiler (and Z3!). Keywords `requires`, `ensures` and `invariant`. DPLL in Dafny was 3200 lines. Main roblem was speed; setLiteral (unit propagation0 took 630 seconds to verify: total 15 minutes. Admittedly much slower than production SAT solvers, but that's not the point. For example, currentlyuses true $\mathbf{Z}$not `int` etc.

See [AC19].

2.4 Gröbner Bases with Reduction Machines: Crăciun

Classic reduction says that you should reduce head terms, but in fact you can reduce any. Indeed, it may be as quick. Hence look to simulate with reduction machines. While this is not as fast as specific solvers, we note that a reduction machine mechanism allows great parallelism.

See [cC19].

Chapter 3

6 September 2019

3.1 Human-robot Interaction: Adina-Magda Florea

Assistive robots etc.; Ambient Intelligence, Telepresence, Humanoid robots. Service robots (pictures in shops). Started with Nao, now Pepper at Tiago. There is a challenge: diversty of robotic units.Nao and Pppr use MAOqi, Tiaho uses ROS — Robotic Operating System. Thwre's a large community growing around ROS. Topic-based publsh-subscribe model for adding new nodes. But there's alck of ROS-based rameworks for behavoural life-cycle for social robots. Hence our AMIRO, a ROS-based framework.

Example 5 *Elderly person ignores mobile notifications to take medicine. The robot searches for the person starting from last known location. After positive identification, can interact with person. There's a planning module, also vision, speech and navigation modules. Naviation functionality includes building a map of surrounds. Visionhasvideo camera and depth sensor, 20 FPS and* 640×480 *and* 320×240 *respectively. People detected trigger extra operations. Use YOLO for obect detection. Object tracking based on SORT. ResNet is used for segmentation. Speech in Englist and Romanian.* `wit.ai` *for speech understanding, but working with Romanian Academy on a Romanian system. The rbot has a map of the laboratory and surrounds. Recognises people in the lab and can deliver mesages to people.*

Video shows robot interacting with multiple people.

3.2 Review of Algorithms on Symbolic Domains: Watt

3.2.1 Computer Algebra versus Symbolic Computation

We have become so good at looking at specific domains than we have forgotten the general meaning of “symbolic”. Once you put a variable into an exponent they fall apart. We want to allow ring operations, such as $x^{n^2} + y^m$, but not $x^{\mathrm{lcm}(m,n)}$. Also allow any integer values in exponents (restricting to non-negative causes a lot of extra bookkeeping). Hence really Laurent polynomials. $R[n_1, \ldots, n_p; x_1, \ldots, x_v]$ is our domain.

Theorem 4 *$R[n_1, \ldots, n_p; x_1, \ldots, x_v]$ is a UFD iff $R[x_1, \ldots, x_v]$ is.*

Not entirely trivial: $x^2 - y^{n^2-n} = (x - y^{(n^2-2)/2})(x + y^{(n^2-2)/2})$. Solution: put exponents in a binomial basis, then geneate new variables for each $x_i^{\left(\begin{array}{c} n \\ k \end{array}\right)}$. There are issues with blowup for $x^{m^{1000}}$ etc,

Theorem 5 (Polynomial Decomposition) *Usual decompositions plus a fixed number of special cases.*

3.2.2 Matrices

$A_{i,j} = \begin{cases} 0 & i > j \\ c_{j-i} & \text{otherwise} \end{cases}$ for example.

But with n symbolic limits, we seem to have 2^n options for $v_i < v_j$, whereas they can’t all occur. Define a support function $\xi(i, k, l) = 1$ iff $i \in [k, l)$, -1 iff $i \in [l, k)$. Then we can “pretend an order” on the index variables. This makes vector addition $O(n)$ rather than $O(2^n)$.

To get to matrix multiplication, we need a similar trick. But “multiplication by 0” isn’t invertible.

3.2.3 Joke

Example 6 (Joke) *2 people enter a house, then three leave. Physicist “error in initial measurement”, Biologist “must have reproduced”, Mathematician “if one more goes in, then the house will be empty”.*

Need hybrid sets, where cardinality $\in \mathbf{Z}$, whereas multisets have $\in \mathbf{N}$. Then have to have genealised partitions.

3.3 iAn Attempt to Enhance Buchberger's Algorithm by Using Remainder Sequences and GCD Operation: Sasaki

PRS is basically univariate, whereas GB multivariate: water and oil. Fundamentally, GB are doubly-exponential (JHD: isn't this a feature of the problem, rather than the algorithm?). So what I would like to do is append polynomials "close to the answer".

Define a $\hat{P}_k := P_k / \gcd(\mathrm{cont}_x(A_k), \mathrm{cont}_x(A_k))$, which isin the ideal.

F is healthy if

1. each of the main variables $x_1, \ldots, x_n$ can be eliminated,
2. none of $u_1, \ldots, u_k$ can be,
3. and the ideal F doesn't split into disjoint GB [?in the x_i and u_j respectively].

Then $GB(F) \cap K[\mathbf{u}] = \langle \hat{s} \rangle$. Hence an algorithm by repeated PRS in the x_i. Shows example where GB has 30,31 ... digit coefficients [but a $yu + \cdots$ polynomial], but the PRS has much smaller ones [but $yu^{14} + \cdots$].

Next step is to cajole the leading cefficient ideal.

3.4 Feature Extraction using Legendre-Sobolev Representation for Handwritten Mathematical Characters: Alvandi/Watt

Recap on handwriting: many people over the years. Our notation was developed for the pen. Note that this also supports editing. Trying to interpret digitised samples of ink, which has problems. We will try to recognise characters as curves $(x, \lambda), y(\lambda))$ rather than as point samples. λ could be time dynamically, but we use arc length, which can be done on static samples. Also (less) resolution dependent. Initailly used Chebyshev (2009). Problem is that low RHS doesn't mean same symbol, as corners needn't be in right place. Also interested in prediction.

Least squares is really minimising a variational integration, which is great when we have orthogonal polynomials. So we need expansions in terms of LS polynomials. But we don't have the usual 3-term recurrence. New theorem about computation. Large matrix (which can be precomputed, but depends on μ, the derivative mixing factor). Similar theorem for GCDs.

Q

A

Q μ?

A Yes, one tunable parameter: $\frac{1}{8}$ works well (but might need tuning for

3.5 Source code vulnerabilities detection using loosely coupled data and control flows: S. Zaharia

Example of CWE23 (Path traversal) in C++. THere are some such, but for given languages only. Our strategy is to uselexical similarities between langauges, e.g. R and D have similarities. So we train our detector on C and Java, where these is a large library of samples, and then apply to new text in new language. MARFCAT is interesteding — “listens” to the source code after a sonic transform — fairly langauge-independent, but alas doesn’t localise the vulnerability. So CWE23 has various keywords, such as `fopen`. We are interested in propagation between variables: not name or type of variables. Flatten the keywords, so all process-spawning commands get the same code.

Use NIST’s database of vulnerable samples. CWE23 has 5474 vulnerable, 100K not, etc. Use $F_1 = \frac{2pr}{p+r}$ where p=precision, r=recall. NN(65,2) and DecisionTree were the best. Trained on C then ran on the Java example (Recall good on the one example, but precision dropped markedly) . Length of code vector is a tunable parameter: use 10, and have an example where a vulnerability is not detected (would need 11)

3.6 Applications of Equivalence Algorithms in Software Design: Alina Andreica

One application was solving currculum equivalence, when students drop out and re-enrol. Solved using “canonical representative” approach. Also looked at applications in data interchange (e.g. student mobility). In that case we could place the canonical database into the cloud as a reference. Also applications in e-learning. Also (joint work with Coimbra) business intelligence applications.

So we all know about equivalence relations. Extend these to sets: $e1 \approx e2 \Leftrightarrow \cdots$. So we want to take the canonical representatives of each element. Extension of pattern matching principles is similar, and again more efficient.

Our implementation is at a database level, with attributes in columns. With 20 or 100 activities per module, we see speedups of $\times 20$ and $\times 80$.

Applications to “association rules in expert systems” worked when we had equivalence relations, e.g. “has the same treatment”.

If two systems have already communicated in the cloud, then the network can grow by transitivity.

3.7 The Science, Art and Magic of Constrained Horn Clauses: Gurfinkel/Björner

AG comes rfrom Model Checking, NB from SAT. Model Checking can be reduced to SAT in contrained FOL. Named after Alfred Horn.

$$\begin{aligned} \forall x \cdot x \leq 0 &\rightarrow P(x) \\ \cdots &\rightarrow \cdots \\ \forall x \cdot P(x) \wedge x \geq 10 &\Rightarrow \text{false} \end{aligned}$$

CHC is a FOL formula

$$\forall V \cdot (\phi \wedge p_1(X) \wedge \cdots \wedge p_n(X)) \Rightarrow h(X)$$

where p_i and h are uninterpreted predicates.

Example of a resolution disproof, which is also a counterexample generator. Many references to Horn clauses in verification. Spacer is the key engine, with frontends like SeaHorn (C/C++) etc. Note that it's part of Z3.

Definition 1 *It is a Craig interpolant between A and B assuming $A \wedge B$ is false.*

1. $A \Rightarrow Itp$
2. $Itp \wedge B \Rightarrow False$
3. *uninterpreted symbols of Itp are those that are shared between A and B*

Example (using Z3) to generate Craig Interpolants: 12 lines of Z3. `https://notebooks.azure.com/arie-gurfinkel/projects/spacerexamples/html/Synasc2019.ipynb`. Also of a transition syste,

Note that a lot of what we are doing is "backwards induction" from the goal. Program is similar to previous, with a few inversions.

Also an example of 2-induction. The (counter-)example is mmore complicated, because of the 2-induction.

Note that he can also do arrays [unbounded size] and quantifiers. Almost always insoluble (model counter automata), but finite cases are often soluble. Various search strategies: SAT (if bounded).

"Art" is generating the right encoding. Changes as underlying solver changes. The Science is the termination when decidable.

Q "Mostly Horn"?

A Rybachenko solves the Horn part first, then looks at the non-Horn part.

NB Z3 has different engines for Horn and non-Horn problems.

Q Why "Z3"?

A Lots of previous systems with '2' in name.

Q Nonlinear?

A–NB Partial linearisation, or Partial CADs

3.8 Portfolio SAT and SMT Solving of Cardinality Constraints in Sensor Network Optimization: Kovásznai

Sensor Device Networks, energy-limted, so scheduling to maximise T the lifetime (time is discretised). n sensor nodes, lifetime L_i, rage r_i, distances $d_{i,j}$ from sensor i to target j. $w_{i,t} = T$ iff node i is awake at time t. $\forall i \sum_{t=1}^{T} w_{i,y} \leq L_i$. Unroll (enumerate) the $\forall i$, then these are boolean cardinality constraints. Also each target is covered all the time. May also have "evasive" constraints: a sensor must not be awake for more than E_i slots running. Also "moving target" [bad name] consraints: the same sensor mostn't cover a target for more than M_j consecutive slots. Hence use OMT solvers. OptiMathSAT, Z3 and Solver: OptiMathSAT was the best as problem difficulty grew.

These work by increasing T, throwing problem at SMT, until they find UNSAT. The bottleneck is solving the SAT instances. So tried using SAT via various encodings of the arithmetic. MiniCARD is an extension of MiniSAT that natively supports cardinality constraints. PySAT and PySMT are good interfaces to multiple solvers. Also parallelism.

20 instances in each benchmark set. 2 density groups [40,50]% [60,70]% and [80,90]%. Evasive on/off; Movin on/off. In one case [60,70]%, all on: Puli was best OMT at 75sec; MiniCARd 2.8, MiniSAT with Seq-Counter coding was 3.5sec, Glucose+Seq 7.6 seconds. Others much worse. At [80,90]% the SEQ did muct better than others.

MiniCARD+Z3 is the best, and outperforms all OMT by 1-2 orders of magnitude. MiniSAT+SEQ encoding is a good alternative. `https://iot.uni-eszterhazy.hu/en/research/tools`.

3.9 Superposition Reasoning about Quantified Bitvector Formulas: Damestani

Based on VAMPIRE.

$$\forall x_{-}[n] \exists y_{[n]} \forall z_{[n]} \exists u_{[n]} x_y = z \times u.$$

Any system can solve with $n = 4$, ut many find 32 hard. Bitvectors is a multi-sorted theory, wih 2^n constants for $[n]$. So add a (structured) class `bv4`. Need to add commutativity, interactions between $\leq$ and : (concatenation). Could solve [2018PreiverCAV]. But weaker than many SML like Z3.

Chapter 4

7 September 2019

4.1 Cellular Automata Applications: Andreica

Key idea is local interactions only. For example, 2D automata. Problem is local $\Rightarrow$ global for behavour, but also $\Leftarrow$: what local rules gave rise to this global hehaviour? How do we get convergence to all0/all1 depending on initial density? Topologies: lattice, or network. Have looked for various topologies, via an evolutionary algorithm. Went back to lattice topology, but added "far neighbours".

Image recognition: depends on tasks.

4.2 Shapley Value and Extremal Optimization for the Network Influence Maximization Problem: Képes

$$\phi_i(v) := \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N|-|S|-1)!}{|N|!}(v(S \cup \{i\}) - v(S)$$

amount player i can add to coalitions. Tried on various publication data sets, with 100s to 1000s of nodes. 1 runs for each sets of parameters, 5000 steps, and generally 30 cascades.

Wilcoxon sign-rank test showed SVEO to be best (once joint-best) in every case. But the running times were greater. Also different data sets wanted different parameter values.

4.3 Protein Folding Simulation using Combinatorial Whale Optimization Algorithm: Sima

We have a hydrohobic-polar model. CWO: compare our point with a random point. Preliminary results were promising.

4.4 Population distribution dynamics in Genetic Algorithms with High-Probability Mutation: Criotoru

Continuation of previous SYNASC. [2014] High-probability mutation "$\approx 95\%$": but in the binary context this leads to a fliping behaviour. Ths is an extension of [2015]. We are more numerically stable than cascade sequences.

We are computing average, rather than median, chromosome on each generation, and measure how much the average changes across generations. Ask where differences arise between generation at each locus: this measures movement between generations. Inour experiments we used 0.001, 0.01, 0.05,0.5, 0.6 and 0.95. as mutation rates.

4.5 Towards Automated Quality Assessment Methods in Algorithmic Music Composition: Sulyok

End of a 5-year PhD journey. Actually looking at similarity, rather than quality. Initialisation, phenotype rendering, then corpus-informed phenotype evaluation, and next generation. Note that, unlike most, the corpus doesn't influence the start point. Our complex model ws based on MIMI, 7-bit values (more than a piano!) etc. The made a reduced model [Solyoketal2019a]. Two corora: Bach (toocomplcated) and Hungarian folk songs. Various fitness measurements in literature.

4.6 Data driven sales prediction using communication sentiment analysis in B2B CRM systems

Really about odds of closing. Note B2B takes longer, often multiple decision makers. Les researced, becaus eless transactions $\rightarrow$ less data, plus harder to obtain. CRM used to track sales process. Looked in particular at ERP sales. Variables included #licences sold, #days sales cycle etc. Three different feasure section algorithms agreed that "#NeutralSentences" [in the CRM log] was top.

We now have a second dataset to analyse, and need to convert this into a decision support system.

Q are your data skewed?

A 123 won,153 lost, so not terribly skewed. Muchmore data on won deals.

Q

A

4.7 Closing

Last date at IEE 13 December. Revised versions to be uploaded as a new version (also with response by replying to the message) by 15 October. Final decisions 15 November.

SYNASC 2020 will be 1–4 September 2020. Then 2022.

Theorem 6

Bibliography

[AC19] C.-C. Andrici and Ş Ciobâcă. Verifying the DPLL Algorithm in Dafny. *EPTCS*, 303:3–15, 2019.

[BFS04] M. Bardet, J.-C. Faugère, and B. Salvy. On the Complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations. `http://www-polsys.lip6.fr/~jcf/Papers/43BF.pdf`, 2004.

[BLB10] R. Brummayer, F. Lonsing, and A. Biere. Automated testing and debugging of SAT and QBF solvers. *International Conference on Theory and Applications of Satisfiability Testing*, pages 44–57, 2010.

[cC19] G. Șurlea and A. Crăciun. Gröbner Bases with Reduction Machines. *EPTCS*, 303:61–75, 2019.

[HS95] M. Huber and B. Sturmfels. A Polyhedral Method for Solving Sparse Polynomial Systems. *Math. Comp.*, 64:1541–1555, 1995.

[Lad76] J.D. Laderman. A Non-Commutative Algorithm for Multiplying 3×3 Matrices Using 23 Multiplications. *Bull. Amer. Math. Soc.*, 82:126–128, 1976.

[New14] C. Newcombe. Why Amazon chose TLA+. *International Conference on Abstract State Machines*, pages 25–39, 2014.

[Smi13] A.V. Smirnov. The bilinear complexity and practical algorithms for matrix multiplication. *Computational Mathematics and Mathematical Physics*, 53:1781–1795, 2013.

[Ste04] H. Stetter. Numerical polynomial algebra. *SIAM*, 2004.