Panel Discussion Slides FM@Scale Workshop

Eric Smith

Kestrel Institute

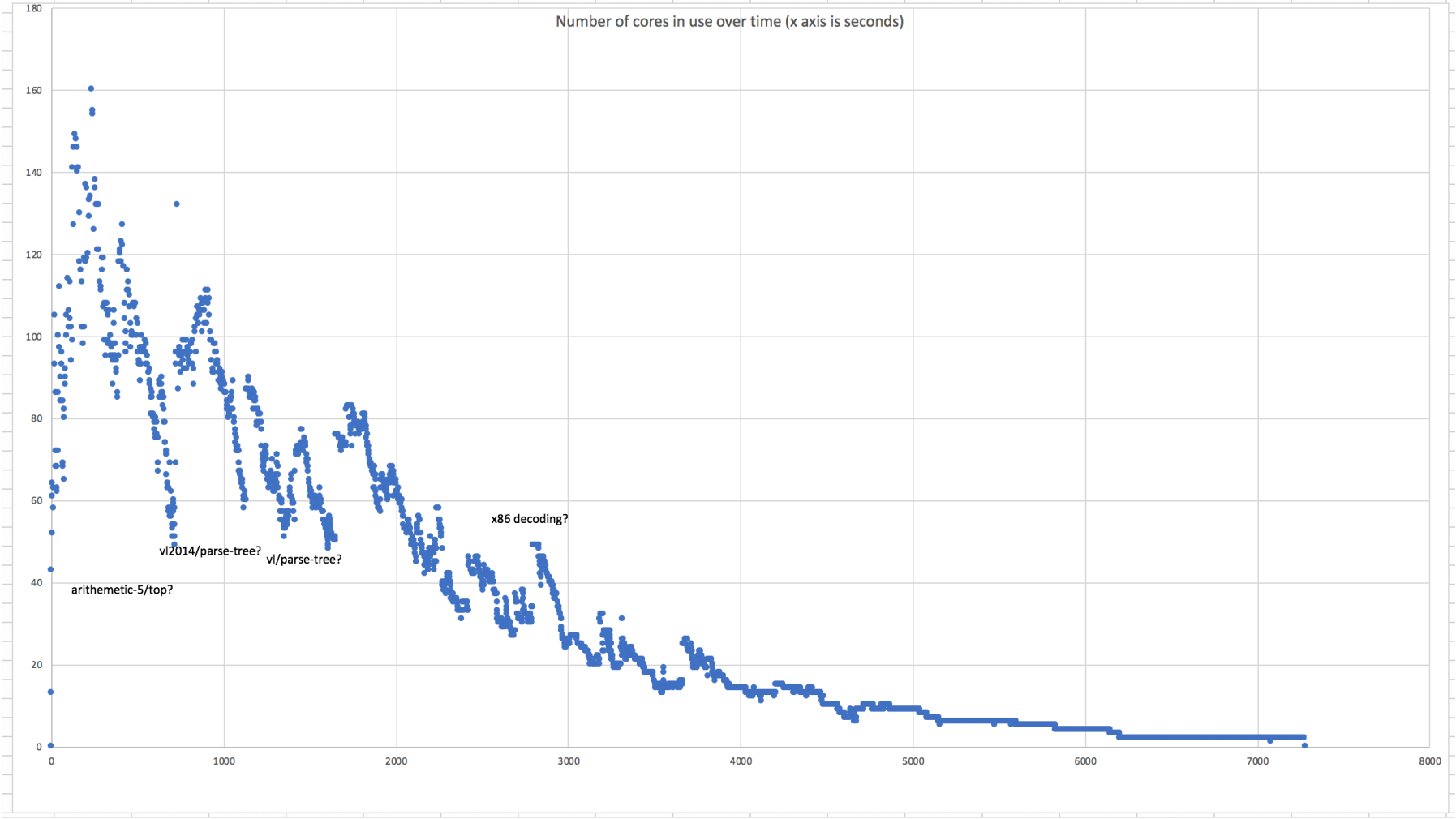
October 9, 2019

SRI, Menlo Park, CA

A Large Formal System: ACL2 and its Community Libraries

- ACL2 (together with its libraries) contains:
  - ~8,000 files
  - ~124,000 definitions
  - ~150,000 theorems
- Kestrel adds
  - ~1,200 files
  - ~18,000 definitions
  - ~19,000 theorems
- Both repos include lemma libraries, tools, and many worked examples
- This all takes quite a few CPU hours to certify (see next slide)
- Frequent changes
  - https://github.com/acl2/acl2 has ~13900 commits in ~5 years
  - Kestrel's main ACL2 repository has ~13500 commits in ~6 years
  - Both see several commits per day
- We keep all this stuff working continuously.
  - Prevents bit-rot
  - Test changes and detects errors, bad decisions, bad rules

Numbers of Cores in Use During a Build of ACL2's Libraries

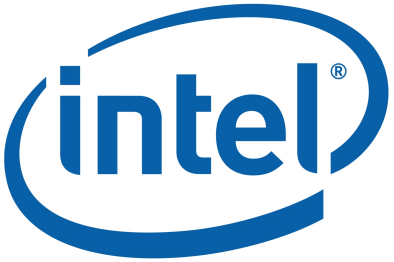


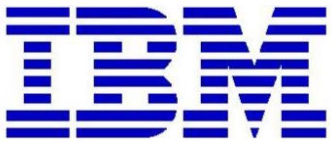
Things that Enable Scale

- Lots of engineering on the prover over 30 years
  - Handle huge industrial designs
- Certificates
  - When ACL2 processes a file, it stores a "certificate"
  - When including a certified file, the proofs are *not* replayed
- Automated regression testing
- Lots of cores
  - My testing uses a 112-core machine
  - Many large Axe / Android / MUSE proofs take ~1hour each
- Good documentation
- Avoiding name clashes
- Working with several simultaneous copies
  - On git, people push to one of four testing branches
    - When any testing branch certifies, the changes are automatically pushed to master
  - I keep ~8 copies of things, so I can test various change sets independently.
- Striving for robustness
  - Avoid brittle proofs hints that depend on, e.g., subgoal numbering

ACL2 Theorem Prover (github.com/acl2)

- Automated, industrial-strength theorem prover
- Used by many companies





Axe: A more scalable rewriter

- Axe is based on ACL2 and includes a rewriter, prover, and equivalence checker
- Key difference: terms are represented as structure-shared DAGs
- Makes symbolic execution much faster
- Crucial for unrolling crypto code (for equivalence checking)

- The Axe Rewriter operates directly on DAGs.
- Have to approximate the context of overarching IFs (may be exponentially many paths to a node).

Example: Blowfish encryption with 448-bit key

```
Showing info for DAG:

220723 unique nodes

690472289016426069683854202738299081803389981811518455131424817761284028924957498204253097051761912907629066455088174920170637969571096322221118078174969070584521423840068628130995627773378097640572862012360973483633027155373933735236936963737219319982842674009574544272057435559983579252426501062582876771159905190782417328523620544632162935407712818305297168171532679565275290969631720570949317096144196977593991231760551084974147622358219320242896679591414268145179672954942133448392457654244648782104045963400219322382752310196025846369353026179357101494554203491928293279857765913892421891949471443874426625042134794770552577650792446363255148701176283932799278095691052854453808409851883803805439628450495469756883134922871102598299196656932990350275423252906000836893896680320058886303041543003337237520596198490848487759644438746475918620712439615600539950383460901315099858135316656188466817424413730955210547684198434122403870946276440924528492041469859371480812318507057570683013033879479716122774362891817965525105130034880199825415662072537093175347855415533137603280045893537011636213930765444811841686865291771092812299901644959988473285508688690004843270444590893803968571348669779160732059779085195779065238051517416898437761934179503391766843298176409197303748158124387425712416143364476707263547362236159303853410439060223303407501501294035713602371232507041016936374989802337837407175453049011718890334320605129084541324401523647169808406079071913377692479639975291567226258273626028189688164095171532758334884835259133179132143616438185911916454636216257941130723113835514574168736468015459352475417828446085504637171629927859236098589273297526894448885991498020756628985261189230284172066154892855313814532572622618601066644738576484380466849881897327756880251945969732274049647214393080671981829742655284234057433534520463870623639543413114588509322533580802638036663037900154971776834586244637625814499226874818758112300313249081093154721676825639127996920513029076031755238791011131625952971177799977743771066606229330281035921682111583809633217970811057957947127390841088588882649467621914108524590027852288131173295880945966188095748454539657236625488914111397088464836319749260680378115050574540401767742810200314852777258760650912348048950007838152718665140818275250266314328379686653706630136966480263475905669480191801377989758782592850520590867336351581514798068226529597924727704064836328322896634890702319688805203852604061708556339642979458262456294824806130508155268690704354432667990975078075279781945689557600272219130799862700806986795055804416071700359595734437779877164578763725280055399295107698609099030527984498068824219783681068924882086989809475493772649970904795695820487744355789320591312757525662228681228285035831180857309927517765132251928387849766391239470317398013010832118711939220906312481392796878812682617661407326282856804004866658066589328478553248225996193560683777975966465687190196812356591261198059118242548444222794708318116990160665803604691403344584489474424037732946104191888237067443790198549607889524550092824501980196171452702614683459502646173347094089204061758093394068827724073829472272582603824935670585708558153496313480930512561202085110293148777138259224014285160689250471216550547775353633251921697575359121052709012048874772876030755658755661568120302681295639468377106275856461351901556112231942057712160413152779885752487262855746689308023630157068045529415272053877877673169333684094537293940918427121658571907855406732463952784597274310033096981635674361674568385197178126515708467705120507552155790807127224174378139329216099644452395883831105178832620421316856986332651656811798706111247460638246759437703727390067670807244035546775552742392764678728502963480118113088241428801025022071881845153889054324717116284165709843640763627586770872531607280507727802306318826802112281941840949207479845198793270066497345171435674947164017906048425651346895739032769689650332405791791891994918170785283241462687193646549116728965158663568839855377710733550027670383427039300465901103364132861733921291189857842622210903550775930681537615144694136418514320221659524746561845595899604352224544887127476063372737697437619038914397213154035641402389954592278790362439904763130369295451126568408311043513131064095232581303554657198569173294877033637025265393606326825152905485338146487378997485094302294138488139044917570782530473177580603090441276695697966552766236852006619214929791626948980394748373764914729998747289947657754235053406021551148438454267993301424713587125095305437402884844538928279013080793411922443609417815958485735791797808258799045045745274403219653252914480208850709945724817303452253084419744424778774793198292478354262658631178581390686293390769174867566841086386267022297367581650651860721756313966581668191736234416846178821008176029154268792453452631058841521059927726378147026598652155746850103286603046858061211839065695494949775545237936886585973593860294894108611640620640578577929510518439794644684734698274966532660433272008411348945604012370463386948055012244185056995724257025540993474694108898032094922318967987863927191231609528842163973706564638043049281622402487511416165151
```

total nodes

Example: Blowfish encryption with 448-bit key

```
Showing info for DAG:

220723 unique nodes

690472289016426069683854202738299081803389981811518455131424817761280963222
211180781749690705845214238400686281309956277733780976405728620123605743555
998357925242650106258287677115990519078241732852362054463216293540773991231
760551084974147622358219320242896679591414268145179672954942133448393571014
945542034919282932798577659138924218919494714438744266250421347947705188380
380543962845049546975688313492287110259829919665693299035027542325299644438
746475918620712439615600539950383460901315099858135316656188466817424808123
185070575706830130338794797161227743628918179655251051300348801998254481184
168686529177109281229990164495998847328550868869000484327044459089384179503
391766843298176409197303748158124387425712416143364476707263547362213749898
023378374071754530490117188903343206051290845413244015236471698084065833488
483525913317913214361643818591191645463621625794113072311383551457417526894
448885991498020756628985261189230284172066154892855313814532572622612143930
806719818297426552842340574335345204638706236395434131145885093225331230031
324908109315472167682563912799692051302907603175523879101113162595297957947
127390841088588882649467621914108524590027852288131173295880945966185745404
017677428102003148527772587606509123480489500078381527186651408182755878259
285052059086733635158151479806822652959792472770406483632832289663488155268
690704354432667990975078075279781945689557600272219130799862700806986090
990305279844980688242197836810689248820869898094754937726499709047952751776
513225192838784976639123947031739801301083211871193922090631248139276193560
683777975966465687190196812356591261198059118242548444222794708318117901985
496078895245500928245019801961714527026146834595026461733470940892041348093
051256120208511029314877713825922401428516068925047121655054777535362681295
639468377106275856461351901556112231942057712160413152779885752487202939409
184271216585719078554067324639527845972743100330969816356743616745689964445
239588383311051788326204213168569863326516568117987061112474606382401811308
824142880102502207188184515388905432471711628416570984364076362758673270066
497345171435674947164017906048425651346895739032769689650332405791797107335
500276703834270393004659011033641328617339212911898578426222109035502454248
712747606337273769743761903891439721315403564140238995459227879036329856917
329487703363702526539360632682515290548533814648737899748509430229412006619
214929791626948980394748373764914729998747289947657754235053406021557934119
224436094178159584857357917978082587990450457452744032196532529144805863117
858139068629339076917486756684108638626702229736758165065186072175631521059
927726378147026598652155746850103286603046858061211839065695494949717946446
847346982749665326604332720084113489456040123704633869480550122441850952884
216397370656463804304928162240248751141616515

total nodes
```

```
((220722 BV-ARRAY-WRITE '8 '8 '0 220705 220721)
 (220721 BV-ARRAY-WRITE '8 '8 '1 220704 220720)
 (220720 BV-ARRAY-WRITE '8 '8 '2 220702 220719)
 (220719 BV-ARRAY-WRITE '8 '8 '3 220700 220718)
 (220718 BV-ARRAY-WRITE '8 '8 '4 220717 220716)
 (220717 SLICE '31 '24 220698)
 (220716 BV-ARRAY-WRITE '8 '8 '5 220715 220712)
 (220715 SLICE '23 '16 220714)
 (220714 BVXOR '24 220666 220713)
 (220713 BVXOR '24 220696 2810))
 ...
 (10 . KEY2)
 (9 . KEY1)
 (8 . KEY0)
 (7 . IN7)
 (6 . IN6)
 (5 . IN5)
 (4 . IN4)
 (3 . IN3)
 (2 . IN2)
 (1 . IN1)
 (0 . IN0))
```

Scaling FM to more widespread use

A general approach: Inject formal into a standard process to make things a little bit better.

- Strive for complete automation.
- Make the "spec" easy to express, or spare the user from writing it.

Concretely (a new start at Kestrel):

- Programmers can write unit tests.
- A typical test covers a single input.
- What if we could replace a test with a little proof?
  - Just do slightly better: cover a set of inputs
  - Tool proves that all inputs in the set produce good behavior, or it finds a counterexample.
  - Tool achieves automation by using symbolic execution (seems okay to restrict to small inputs) and SMT solving.

- Don't compare this to a full proof and feel bad about it.
- Compare this to a unit test and feel good about it!