Selecting the Selection

Giles Reger Martin Suda Andrei Voronkov

University of Manchester

27th June 2016

Finding Proofs (quickly)

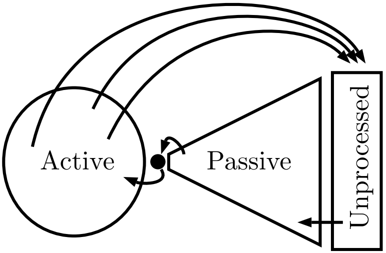
- This talk/paper is about what kind of literal selection improves our chances of finding (quick) solutions
- Based on the Vampire theorem prover

- **Quality-Based Selection.** Select high-quality literals that lead to the fewest new clauses
- **Lookahead Selection.** Estimate the number of new children and select the smallest
- **Incomplete Selection.** Drop the completeness condition to remove restrictions on the above heuristic

- **Conclusion.** A mix of all of these solves the most

Saturation-Based Proof Search

- Saturate a set of clauses with respect to an inference system



- Want to stop clauses space growing too quickly, either
  - Remove what has already been generated (e.g. subsumption)
  - Restrict what is generated in the future (literal selection)

Literal Selection

- Early notion due to Bachmair and Ganzinger
- Use an ordering and selection function to restrict which literals inferences are performed on
  - Select a multi-subset of negative literals in each clause
  - If no selected literals, perform inference on maximal only
  - If selected literals, perform inference on selected literals only
- We consider a more general formulation
- A literal selection strategy is a procedure that assigns to a non-empty clause C a non-empty multiset of its literals.
- Not a function (non-deterministic, depend on context)

The Calculus

Resolution

$$\frac{\underline{A} \vee C_1 \quad \underline{\neg A'} \vee C_2}{(C_1 \vee C_2)\theta}\ ,$$

Factoring

$$\frac{\underline{A} \vee A' \vee C}{(A \vee C)\theta}\ ,$$

where, for both inferences, $\theta = \mathrm{mgu}(A, A')$ *and* A *is not an equality literal*

Superposition

$$\frac{\underline{l \simeq r} \vee C_1 \quad \underline{L[s]_p} \vee C_2}{(L[r]_p \vee C_1 \vee C_2)\theta} \quad \text{or} \quad \frac{\underline{l \simeq r} \vee C_1 \quad \underline{t[s]_p \otimes t'} \vee C_2}{(t[r]_p \otimes t' \vee C_1 \vee C_2)\theta}\ ,$$

where $\theta = \mathrm{mgu}(l, s)$ *and* $r\theta \not\succeq l\theta$ *and, for the left rule* $L[s]$ *is not an equality literal, and for the right rule* $\otimes$ *stands either for* $\simeq$ *or* $\not\simeq$ *and* $t'\theta \not\succeq t[s]\theta$

EqualityResolution

$$\frac{\underline{s \not\simeq t} \vee C}{C\theta}\ ,$$

where $\theta = \mathrm{mgu}(s, t)$

EqualityFactoring

$$\frac{\underline{s \simeq t} \vee s' \simeq t' \vee C}{(t \not\simeq t' \vee s' \simeq t' \vee C)\theta}\ ,$$

where $\theta = \mathrm{mgu}(s, s')$, $t\theta \not\succeq s\theta$, *and* $t'\theta \not\succeq s'\theta$

Completeness

- It is a standard result that arbitrary selection can lead to <u>incompleteness</u>
- Consider

$$p \vee \underline{q} \qquad \underline{p} \vee \neg q \qquad \underline{\neg p} \vee q \qquad \neg p \vee \underline{\neg q}$$

  were <u>all</u> selected literals are underlined
- It is unsatisfiable but saturated
- There is also the standard sufficient condition

  Select either a negative literal or all maximal literals with respect to $\prec$

  from the completeness proof of Bachmair and Ganzinger

Observation

- There is a general insight that a slowly growing search space is superior to a faster growing one, provided completeness is not compromised too much
- It follows that the aim of a selection strategy in our setting is to generate the fewest new clauses

Quality-Based Selection

- We introduce a new notion of selection based on the so-called quality of literals

- A quality preorder on literals $l_1 \rhd l_2$ means that we would prefer to perform an inference on literal l_1 to l_2
- We want preorders that (heuristically) prefer literals having as few children as possible

- Given a quality preorder $\rhd$ define a selection strategy $\pi_\rhd$ that selects the greatest (highest quality) literal with respect to $\rhd$
- Ties are broken arbitrarily but deterministically

Quality Orderings

Unifiability

$l_1 \rhd_{weight} l_2$	$\mathrm{wgt}(l_1) > \mathrm{wgt}(l_2)$	complex structure
$l_1 \rhd_{vars} l_2$	$\mathrm{vars}(l_1) < \mathrm{vars}(l_2)$	$p(f(a), y) \rhd_{vars} p(f(x), y)$
$l_1 \rhd_{top} l_2$	$\mathrm{tvar}(l_1) < \mathrm{tvar}(l_2)$	$p(f(f(x))) \rhd_{top} p(x)$
$l_1 \rhd_{dvar} l_2$	$\mathrm{dvar}(l_1) < \mathrm{dvar}(l_2)$	$p(f(x), f(x)) \rhd_{dvar} p(f(x), f(y))$

Equality and Polarity

$L \rhd_{nposeq} s \simeq t$	L not equality	prolific superposition
$s \not\simeq t \rhd_{nposeq} s' \simeq t'$		prolific superposition
$s \not\simeq t \rhd_{neq} L$	L not equality	equality resolution
$\neg A \rhd_{neg} A'$		default

Completeness

- We complete a selected strategy $\pi_\triangleright$ using the following steps
- N are the literals in a clause and M is the maximal subset

1. **If** $\pi_\triangleright(N)$ is negative **then** select $\pi_\triangleright(N)$
2. **If** $\pi_\triangleright(N) \in M$ and all literals in M are positive **then** select M
3. **If** M contains a negative literal **then** set N to be the set of all negative literals in M and **goto** 1
4. Remove $\pi_\triangleright(N)$ from N and **goto** 1

- The idea is to select the highest quality literal that preserves the completeness condition

- We use both incomplete and complete versions
- Saturation with incompleteness must return Unknown

Another Observation

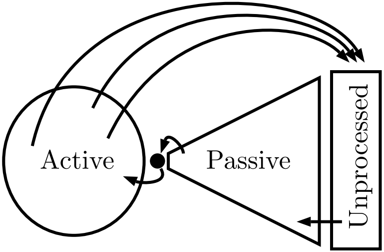
- If we are trying to select the literals that will lead to the fewest new clauses then why not compute this
- Let children(C, l) be the number of children of clause C if literal l is selected in C
- This leads to a new quality preorder where we minimise this value

$$l_1 \rhd_{lmin} l_2 \quad iff \quad \text{children}(C, l_1) < \text{children}(C, l_2)$$

- We can also try maximising this value (expecting it to be bad)

$$l_1 \rhd_{lmax} l_2 \quad iff \quad \text{children}(C, l_1) > \text{children}(C, l_2)$$

Given-Clause Algorithm and Term Indexing



- The set of active clauses is stored in indexing structures
- Allow for efficient computation of unifying and matching literals/terms
- Roughly, an indexing structure for each inference

Estimating Children

- Let $\mathcal{T}_1, \ldots, \mathcal{T}_n$ be a set of term indexes
- An estimate for children(C, l) can then be given by

$$\text{estimate}(l) = \Sigma_{i=1}^{n} |\mathcal{T}_i[l]|.$$

- Step through indices in a fail-fast fashion

- This is an over-estimate
    - Side conditions are not checked
    - Children may not necessarily survive retention tests

  but checking such things would be too expensive

- This can be extended to inferences that do not rely on inferences e.g. equality resolution

- Select literals in given clause when it is chosen

Completeness

- As comparing literals is now much more expensive we modify the previous approach

- If there are no negative literals then select all maximal literals
- If there is a single maximal positive literal choose between all
- Otherwise choose between all negative literals

Experiment

- Used 23 selection strategies implemented in Vampire
- Ran on 11,107 problems
    - non-trivial problems from FOF and CNF TPTP 6.3.0
    - Trivial means having a rating 0
    - Excluded unit equality problems
- Used default strategy with
    - Time limit of 10 seconds
    - Age-Weight ratio of 1:5
    - Try with splitting (AVATAR) on and off
- Ran using StarExec

The Selection Strategies (Vampire)

- Selection 0 selects everything
- Selection 1 selects all maximal
- Six selection strategies based on quality preorders

$$
\begin{aligned}
\rhd_2 &= \rhd_{weight} \\
\rhd_3 &= \rhd_{noposeq} \circ \rhd_{top} \circ \rhd_{dvar} \\
\rhd_4 &= \rhd_{noposeq} \circ \rhd_{top} \circ \rhd_{var} \circ \rhd_{weight} \\
\rhd_{10} &= \rhd_{neq} \circ \rhd_{weight} \circ \rhd_{neg} \\
\rhd_{11} &= \rhd_{lmin} \circ \rhd_3 \\
\rhd_{12} &= \rhd_{lmax} \circ \rhd_3
\end{aligned}
$$

- And their complete versions $10\star$ e.g. 1002

The Selection Strategies (Other)

SPASS Inspired

- 20: off. Select all maximal
- 22: always. Select negative with max weight, otherwise 20
- 21: several. Select unique maximal, otherwise 22

E Inspired. Always falls back to selecting all maximal

- 30: neg. Select all negative
- 31: var. Select a negative equality between variables
- 32: small. Select negative with minimal weight
- 33: diff. Select neg that max diff of weight of lhs and rhs
- 34: ground. Select negative ground with max difference
- 35: optimal. 34 and fallback to 33

Adaptations are approximate as E's notion of term weight differs

Ranking the Selections

selection	#solved	%total	#unique	u-score	#child (s.o./all)
1011	4718	83.9	156	563.6	4.2 / 9.9
1010	4461	79.3	31	384.1	9.4 / 14.6
11	4333	77.0	26	354.7	6.5 / 13.6
1002	4327	76.9	62	396.1	8.7 / 15.4
10	4226	75.1	8	283.3	9.9 / 14.5
...	...	...	...	...	...
30	3559	63.3	3	204.9	16.6 / 28.8
32	3538	62.9	5	209.8	6.3 / 19.9
0	3362	59.8	8	203.1	35.8 / 48.7
12	3308	58.8	3	183.4	14.0 / 24.5
1012	2532	45.0	5	146.1	13.9 / 30.8

The Splitting Effect (AVATAR off)

selection	#solved	%total	#unique	u-score	#child (s.o./all)
1010	4289	80.0	64	379.8	9.3 / 17.0
1011	4255	79.4	104	412.7	8.5 / 15.0
1002	4207	78.5	45	356.2	7.5 / 18.5
11	4121	76.9	25	292.9	12.1 / 25.7
10	4116	76.8	9	251.7	13.1 / 21.2
...	...	...	...	...	...
32	3482	65.0	2	188.5	7.8 / 31.9
20	3479	64.9	0	182.2	21.7 / 33.3
12	3313	61.8	6	173.8	25.0 / 33.9
0	3279	61.2	24	206.4	59.2 / 83.1
1012	2403	44.8	7	126.7	17.9 / 36.4

Satisfiable Problems

AVATAR on (total 287)				
selection	#solved	%total	#unique	u-score
33	248	86.4	0	24.5
22	247	86.0	0	24.1
11	246	85.7	0	23.4
32	241	83.9	1	23.8
1	238	82.9	0	21.6
AVATAR off (total 207)				
selection	#solved	%total	#unique	u-score
11	195	94.2	0	16.7
4	191	92.2	0	17.1
3	190	91.7	0	16.9
32	184	88.8	0	14.7
35	183	88.4	0	14.6

Impact on Portfolio Solving

selection	Problems solved only using this selection	
	All	Problems solved only by Vampire
11	151	118
1011	78	62
1	62	58
10	55	41
lookahead	278	216
non-lookahead	502	377
complete	824	691
incomplete	229	169

Summary

- Literal Selection matters (see previous slide)
- The heuristic of minimising the number of children seems to work well
- Lookahead selection does a very good job at keeping the clause search space small
- Different (quality-based) heuristics for selection give needed variance
- Dropping completeness can help solve more